Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Ivan Majhen

Mikrokrmilna knjižnica Arduino za Wi-Fi Direct modul GainSpan 1500M in Wi-Fi Direct aplikacija Android

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Ljubljana 2014

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Ivan Majhen

Mikrokrmilna knjižnica Arduino za Wi-Fi Direct modul GainSpan 1500M in Wi-Fi Direct aplikacija Android

DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Mentor: izr. prof. dr. Patricio Bulić
Somentor: doc. dr. Gregor Papa

Ljubljana 2014

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Proučite značilnosti standarda Wi-Fi Direct ter načine vzpostavljanja povezav Wi-Fi Direct. Izdelajte mikrokrmilno knjižnico za Arduino UNO, ki bo krmilila vzpostavljanje povezav Wi-Fi Direct in TCP/IP z brezžičnim modulom GainSpan 1500M. Knjižnica naj omogoča branje in pošiljanje sporočil TCP/IP.

Izdelajte aplikacijo za operacijski sistem Android, ki bo omogočala avtomatično vzpostavljanje povezav Wi-Fi Direct in TCP/IP, shranjevanje lastnosti le teh in branje in pošiljanje sporočil TCP/IP.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Ivan Majhen, z vpisno številko **63000352**, sem avtor diplomskega dela z naslovom:

Mikrokrmilna knjižnica Arduino za Wi-Fi Direct modul GainSpan 1500M in Wi-Fi Direct aplikacija Android

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom izr. prof. dr. Patricia Bulića in somentorstvom doc. dr. Gregorja Pape,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 23. september 2014 Podpis avtorja:

Zahvaljujem se Institutu "Jožef Stefan" za izposojeno opremo in zanimivo temo diplomskega dela.

Svojim staršem.

Kazalo

Povzetek

Cilj diplomske naloge je razvoj mikrokrmilne knjižnice za mikrokrmilno platformo Arduino UNO, ki bo opravljala komunikacijo preko serijskega vmesnika UART z brezžičnim modulom GainSpan 1500M. Modul podpira dokaj nov način brezžične komunikacije Wi-Fi Direct. Z vključitvijo knjižnice v uporabniški mikrokrmilni program je uporabniku omogočena nastavitev parametrov povezave Wi-Fi Direct. Knjižnica bo s pomočjo uporabniških parametrov krmilila vzpostavljanje povezave Wi-Fi Direct s pošiljanjem ukazov modulu. Na podlagi obvestil modula se bo odzivala z zaporedjem ukazov modulu, s ciljem vzpostavljanja povezave Wi-Fi Direct. Po vzpostavitvi povezave Wi-Fi Direct se bo vzpostavila komunikacija s protokolom TCP/IP, uporabniškemu mikrokrmilnemu programu pa bo s pomočjo funkcij knjižnice omogočeno branje in pošiljanje sporočil TCP/IP. Aplikacija za operacijski sistem Android bo omogočala avtomatično vzpostavljanje in shranjevanje lastnosti povezav ter branje in pošiljanje sporočil TCP/IP.

Ključne besede: Wi-Fi Direct, mikrokrmilnik, Android, TCP/IP, UART.

Abstract

Objective of the thesis is to develop microcontroller library for Arduino UNO microcontroller platform that will carry communication via serial UART interface with a wireless module GainSpan 1500M. The module supports a fairly new way of wireless communication, Wi-Fi Direct. Inclusion of library in the user microcontroller program will allow the user to set the parameters of Wi-Fi Direct connection. The library will be using user parameters to control and establish Wi-Fi Direct connection by sending commands to the module. On the basis of the notifications, module will respond with the command sequence to the module with the aim of establishing a Wi-Fi Direct connection. After Wi-Fi Direct connection is established, communication with the TCP/IP protocol will be established and user microcontroller program can read and send TCP/IP messages with the library functions. Application for the Android operating system will enable automatic connection setup, saving and sending TCP/IP messages.

Keywords: Wi-Fi Direct, microcontroller, Android, TCP/IP, UART.

POGLAVJE 1

Uvod

V današnjih časih vseprisotnega računalništva (angl. ubiquitous computing) in interneta stvari (angl. internet of things) se nam kot inženirjem računalništva poraja želja, da sami zasnujemo napravo, ki posamezniku olajša vsakdan. Mobilne naprave, predvsem pametni telefoni in tablice, ki imajo različne zmožnosti brezžičnega povezovanja ter velike zaslone občutljive na dotik, se nam predstavljajo kot najboljša izbira pri krmiljenju in zaznavanju takšne naprave.

Z uporabo mikrokrmilnikov, posebej pa mikrokrmilne platforme Arduino, nam je na enostaven način omogočeno programsko upravljanje z vhodi in izhodi mikrokrmilnika, s tem pa tudi krmiljenje naprav. Vse večja priljubljenost mikrokrmilne platforme Arduino ter potreba po oddaljenem krmiljenju naprav je od proizvajalcev zahtevala razvoj majhnih brezžičnih modulov. Z brezžičnim krmiljenjem in zaznavanjem naprave s pomočjo pametnega telefona se izognemo zapletenemu razvoju in programiranju vhodnih in izhodnih naprav mikrokrmilnika, kot so npr. tipkovnica in zaslon ter fizični povezavi z napravo. Brezžična komunikacija se na nekaterih področjih predstavlja kot edina možna rešitev, s pojavom in hitrim razvojem cenovno dostopnih brezžičnih modulov pa tudi kot najbolj logična izbira.

V diplomskem delu bomo predstavili komunikacijo med mikrokrmilno platformo Arduino UNO, brezžičnim modulom GainSpan 1500M in mobilno napravo z operacijskim sistemom Android. Sama komunikacija poteka z uporabo različnih komunikacijskih kanalov. Med mikrokrmilno platformo in brezžičnim modulom poteka po žicah s protokolom UART, med brezžičnim modulom in napravo Android pa brezžično s standardom Wi-Fi Direct. Prenos, oziroma enkapsulacija sporočil po komunikacijskih kanalih poteka z uporabo protokola TCP/IP. Pri UART komunikaciji je začetek in konec sporočil določen z zaporedjem znakov ki ga definira brezžični modul, pri napravi Android pa operacijski sistem poskrbi za prenos

sporočil do uporabniške aplikacije.

Vzpostavljanje povezave Wi-Fi Direct, branje, pošiljanje in odzivanje na sporočila med omenjenimi napravami je potrebno programsko obvladovati. Razvili bomo knjižnico za mikrokrmilno platformo Arduino UNO, ki bo uporabniškemu programu omogočala avtomatično vzpostavljanje povezav Wi-Fi Direct in TCP/IP ter branje in pošiljanje sporočil TCP/IP. Predstavili bomo razširjeno aplikacijo Wi-Fi Direct, ki omogoča avtomatično shranjevanje lastnosti povezav Wi-Fi Direct in TCP/IP ter avtomatično vzpostavljanje le teh.

V drugem poglavju bomo na splošno opisali brezžični standard IEEE 802.11, v tretjem poglavju pa razširitev standarda, standard Wi-Fi Direct.

V četrtem poglavju bomo predstavili lastnosti vgrajenih sistemov, oziroma mikrokrmilnikov ter najbolj pogosto uporabljane vhodno izhodne vmesnike. V nadaljevanju bomo opisali postopke potrebne pri vzpostavljanju serijske povezave med mikrokrmilnikom in modulom, ukaze za vzpostavljanje povezav Wi-Fi Direct med računalnikom, modulom in pametnim telefonom, ter predstavili način delovanja in uporabe mikrokrmilne knjižnice.

V petem poglavju bomo predstavili aplikacijo za operacijski sistem Android, ki omogoča komunikacijo z mikrokrmilnikom.

POGLAVJE 2

Brezžična omrežja

2.1 Zgodovina razvoja

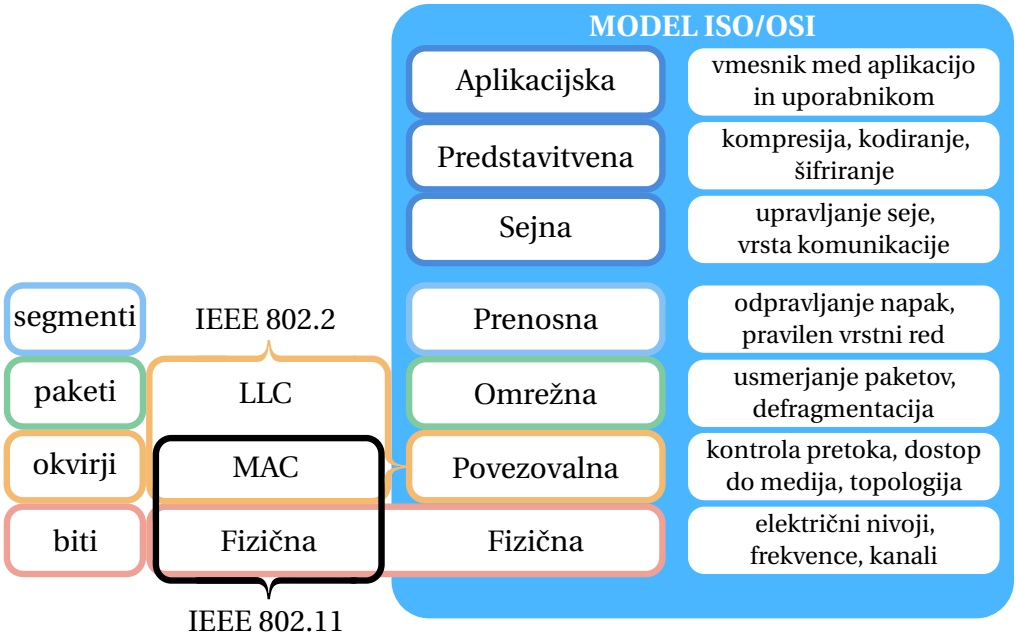
Prvo brezžično omrežje vzpostavljeno med računalniki je leta 1971 ustvarila skupina raziskovalcev iz univerze na Havajih. Poimenovali so ga ALOHAnet in velja za prvo brezžično lokalno omrežje, na kratko WLAN. Centralno vozlišče je oddajalo in sprejemalo podatke na dveh UHF frekvencah s hitrostjo 9600bps. Vozlišče, HP 2100, je pravilno prejete podatke razdelilo na pakete ter posredovalo centralnemu računalniku IBM 360. Težava, ki se ji je bilo nujno potrebno izogniti je bila kolizija, oziroma trk, ki se zgodi ko več postaj sočasno oddaja. Tako so nastali prvi mehanizmi za izogibanje in razrešitev kolizij brezžičnih omrežij – Aloha, Pure ALOHA in Slotted ALOHA, ter prvi MAC protokol za brezžična paketna omrežja. Zaradi neobstoja standardov ter nezdružljivosti obstoječih WLAN omrežij je bila leta 1990 ustanovljena skupina za razvoj standardov WLAN znotraj organizacije IEEE ki si je prizadevala izdelati enoten standard. Tako je bil leta 1997 sprejet standard IEEE 802.11 ki deluje na ISM frekvenčnem območju 2.4*GHz*. Leta 1999 je bila ustanovljena neprofitna organizacija Wi-Fi Alliance s ciljem splošnega sprejetja in promocije standarda IEEE 802.11, spodbujanja pravilne implementacije, ter zagotavljanja skladnosti standarda s pomočjo strogega postopka certificiranja naprav. Danes obstaja približno 5000 certificiranih naprav [7].

2.2 Standard IEEE 802.11 in ISO/OSI model

Referenčni model ISO/OSI predstavlja omrežno komunikacijo kot modularno zgradbo protokolov ki deluje kot celota. Predpisuje vmesnike med posameznimi plastmi in funkcij, ki jih le ti morajo opravljati. Vsaka plast je atomarna in lahko komunicira samo z višjo in nižjo

plastjo, uspešno preneseni podatki pa morajo potovati skozi vse plasti. Zgornje tri plasti (aplikacijska, predstavitvena, sejna) omogočajo podporo uporabniškim aplikacijam, spodnje štiri pa zagotavljajo prenos podatkov. Fizična plast je izvedena v strojni opremi, vse ostale pa v programski. Standard IEEE 802.2 deli povezovalno plast na dve podplasti – podplast logične povezave (LLC) in podplast dostopa do medija (MAC) (slika 2.1). Podplast LLC predpisuje vmesnik do omrežne plasti, podplast MAC pa način dostopa do skupnega medija. Takšna razdelitev ima dve dobri lastnosti:

1. uporabniku in uporabniškim aplikacijam omogoča transparenten dostop do omrežnih storitev,
2. razvijalcem strojne in programske opreme predpisuje le vmesnike na najnižjih plasteh ki jih morajo implementirati pri razvoju omrežne naprave.



Slika 2.1: ISO/OSI referenčni model in IEEE 802 standard

Skupni medij brezžične komunikacije je zrak po katerem se prosto razširja elektromagnetno valovanje. Širjenje signala preko zraka je bolj neodporno na motnje in napake v prenosu kot prenos signala po žici. Število naprav v oddajnem okolju je neznano, okolje nima jasno določenih prostorskih mej in se dinamično spreminja, moč signala je časovno spremenljiva, naprave pa so nezaščitene pred signali drugih naprav. Zato so potrebni drugačni mehanizmi in pristopi k obvladovanju medija, kolizij, kontrole prenosa ter zaščite podatkov.

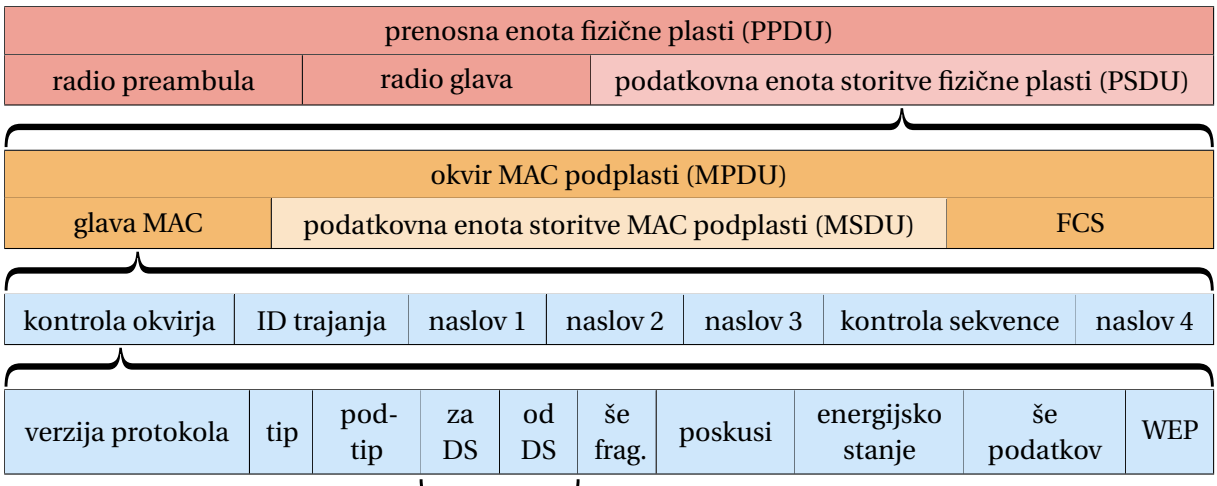
Standard IEEE 802.11 je del skupine IEEE 802 omrežnih standardov ki predpisuje spodnje dve plasti ISO/OSI modela brezžičnih omrežij, natančneje fizično plast (PHY) in podplast MAC. Vsako napravo ki komunicira po standardu IEEE 802.11 bomo v nadaljevanju imenovali postaja (angl. station).

2.3 Podplast MAC

Podplast MAC opredeljuje protokol MAC ki opravlja storitve, potrebne za zagotavljanje zanesljivega mehanizma prenosa podatkov preko glasnega, nezanesljivega brezžičnega medija. Storitve MAC protokola:

- zanesljiv prenos podatkov,
- pravičen dostop do deljenega medija,
- zaščita podatkov.

Na tej plasti definira okvir (angl. frame) kot osnovno prenosno enoto protokola (MPDU). Okvir je sestavljen iz glave MAC, podatkov posredovanih iz LLC podplasti (MSDU) in kontrolne vsote okvirja (FCS) (slika 2.2).



			naslov 1	naslov 2	naslov 3	naslov 4
komunikacija med postajami	0	0	cilj	izvor	BSSID	
od dostopne točke	0	1	cilj	BSSID	izvor	
do dostopne točke	1	0	BSSID	izvor	cilj	
komunikacija WDS	1	1	prejemnik	oddajnik	cilj	izvor

Slika 2.2: Okvirji na fizični plasti, podplasti MAC ter polja glave MAC

2.3.1 Opis poljev glave MAC

Polje ID trajanja je dolžine 16 bitov in vsebuje čas trajanja vsakega polja, pri kontrolnim okvirjima pa vsebuje ID asociacije (AID). Naslovna polja vsebujejo naslove MAC, njihov pomen pa je določen v odvisnosti od stanja enobitnih polj "za DS" in "od DS". Če sta npr. obe DS polji 1, pomeni da komunikacija poteka med distribucijskima sistemoma, če sta 0, pa postaje medsebojno komunicirajo (več v poglavlju 2.5). Kontrola sekvence omogoča koordinacijo okvirjev in fragmentov ter izločevanje napačno ponavljajočih okvirjev. Polja tip in podtip določajo vrsto okvirja. Standard definira tri glavne vrste okvirjev:

- upravljavni (angl. management) opravljajo storitve povezovanja postaj (asociacija, avtentifikacija),
- kontrolni (angl. control) skrbijo za upravljanje in kontrolo prenosa okvirjev (potrjevanje, RTS/CTS),
- podatkovni (angl. data) prenašajo podatke višje ali nižje ležeči plasti.

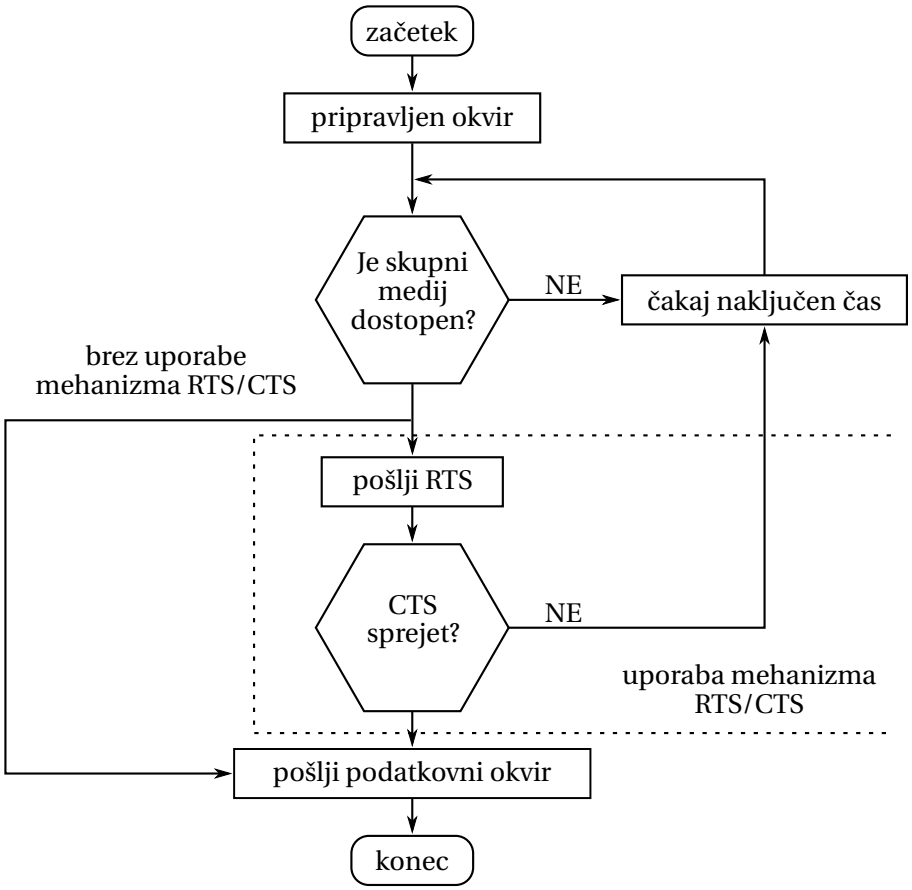
2.3.2 Zanesljiv prenos podatkov

Minimalna komunikacija je sestavljena iz dveh okvirjev, okvirja ki se pošilja od izhodišča do ciljne postaje in potrditvenega okvirja (ACK) ciljne postaje, ki izhodišču potrdi sprejem. Če izhodišče ne sprejme potrditvenega okvirja, se prenos ponovi. Vsak okvir ima kontrolno vsoto (FCS) ki zagotavlja integriteto sporočila. Pred pošiljanjem, postaja iz bitov okvirja s pomočjo zgoščevalne funkcije CRC izračuna kontrolno vsoto. Sprejemna postaja ponovi postopek in če se kontrolni vsoti ne ujemata, enostavno ne upošteva okvirja.

2.3.3 Pravičen dostop do deljenega medija

Prenos okvirjev je v half–duplex načinu, kar pomeni, da mora postaja poslušati deljeni medij in se prepričati, ali ima pravico do prenosa. Pravico do pošiljanja ima, kadar nobena postaja ne pošilja, v nasprotnem se povozi okvir druge postaje. Ciljna postaja takšen prenos obravnava kot motnjo, izhodiščne pa zatorej ne sprejmejo potrditvenih okvirjev in prenos morajo ponoviti. Ponoven prenos postaje morajo opraviti po naključnem času po dostopnosti medija, namreč, lahko bi se zgodilo, da bi spet sočasno poslale okvir. Za preprečevanje takšnih dogodkov skrbi kolizijski mehanizem CSMA/CA. Pri brezžičnem mediju je zaznavanje medija dokaj težavno, namreč, centralna postaja lahko sprejema signale obeh, zunanje pa sprejemajo samo signal centralne postaje – ti. problem skrite postaje. Zaradi tega je znotraj CSMA/CA mehanizma dodan zaščitni protokol RTS/CTS s pomočjo katerega centralna postaja upravlja

prenose ostalih postaj. Ko ima postaja pripravljen okvir in ko je medij dostopen, pošlje okvir RTS centralni postaji, ki ji dovoli prenos z okvirjem CTS. Vse ostale postaje, ki sprejmejo okvir CTS počakajo s prenosom za čas, ki je določen v okvirju CTS (slika 2.3).



Slika 2.3: Diagram poteka kolizijskega mehanizma CSMA/CA z ali brez upravljalnega mehanizma RTS/CTS

2.3.4 Zaščita podatkov

Ko postaje niso v stanju pošiljanja, spremljajo vse okvirje in preverjajo če je naslov MAC enak njihovemu, oziroma če je namenjen njim. To pomeni da lahko berejo tudi podatkovni del okvirja, oziroma uporabniške podatke. Zaradi tega je potrebno podatkovni del zaščititi. Obstaja več zaščitnih tehnik, ki jih IEEE 802.11 predpisuje, spreminjale pa so se predvsem zaradi pomanjkljivosti kriptografskih algoritmov ter nezmožnosti avtentikacije posamezne postaje oziroma uporabnika. Več v poglavju 2.6.

2.4 Standardi 1997/a/b/g/n

Zaradi hitrega razvoja radiokomunikacijskih tehnologij ter potreb po hitrejšem prenosu podatkov se standard nenehno nadgrajuje z novimi različicami. (tabela 2.1).

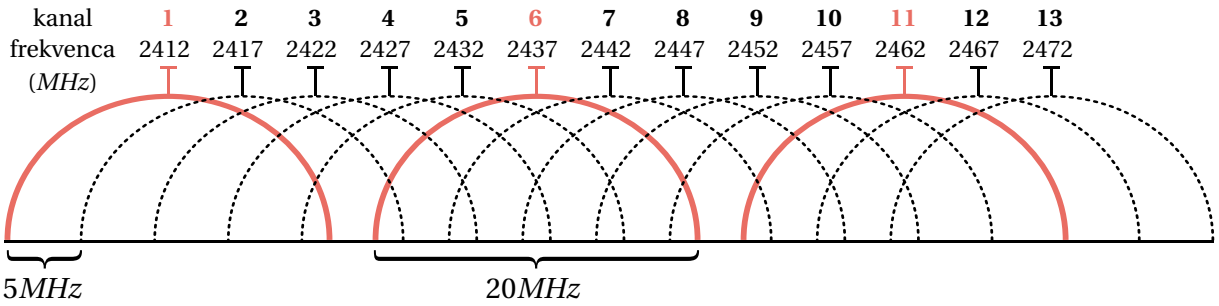
Tabela 2.1: Standardi IEEE 802.11

leto	standard	frekvenca (*GHz*)	širina kanala (*MHz*)	modulacija	največja hitrost	tehnologija antene
1997	802.11	2.4	20	FHSS	2Mbps	
1999	802.11b	2.4	20	DSSS	11Mbps	
1997	802.11a	5	20	OFDM	54Mbps	
2003	802.11g	2.4	20	DSSS, OFDM	54Mbps	
2009	802.11n	2.4, 5	20, 40	OFDM	600Mbps	4xMIMO
2012	802.11ad	60	2160	SC, OFDM	6.76Gbps	beamforming
2013	802.11ac	5	40, 80, 160	OFDM	6.93Gbps	8xMIMO

Standardi 1997/b/g/n so medsebojno združljivi. Novejši standardi frekvenčnega območja 2.4*GHz* so zasnovani tako, da pri komunikaciji s starejšimi uporabljajo starejši standard (angl. mixed mode). Standardi IEEE 802.11g/n uporabljajo drugačno modulacijo, ki jo starejši standardi ne razumejo in jo obravnavajo kot motnjo, zaradi česa bi lahko povzročali kolizije. Rešitev težave je uporaba protokola RTS/CTS, čigavi okvirji se pošiljajo v kompatibilnem načinu (angl. legacy mode) [13].

Frekvenčno območje 2.4*GHz* je v Evropi razdeljeno na 13 kanalov pasovne širine 20*MHz* s 5*MHz* pasovne širine med sosednima. Zaradi relativno majhne razdalje med njimi prihaja do motenj. Pravilno delovanje, brez motenj je pri vsaj 25*MHz* razlike, kar pomeni, da sta kanala med seboj ločena z vsaj petimi kanali. Tako na koncu ostanejo le tri neprekrivajoči kanali: 1, 6 in 11 (slika 2.4).

Vsi standardi frekvenčnega območja 2.4*GHz* imajo enake težave: premalo neprekrivajočih kanalov ter interference z drugimi napravami (mikrovalovne pečice, brezžični telefoni, Bluetooth). Povprečni domet vseh različic standarda IEEE 802.11 frekvenčnega območja 2.4*GHz* je približno 40*m* v zaprtih prostorih in do 140*m* pri optični vidljivosti.



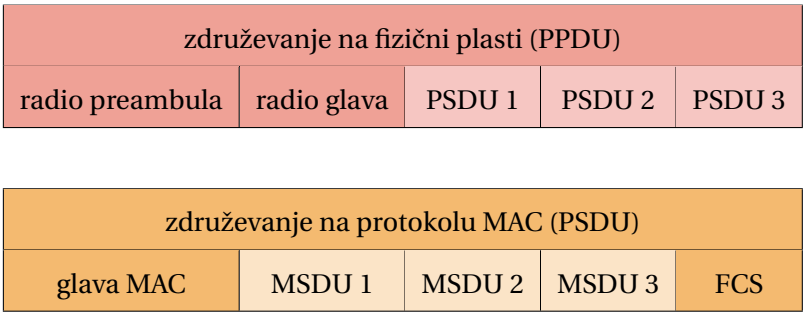
Slika 2.4: IEEE 802.11 2.4*GHz* prekrivanje kanalov

IEEE 802.11n podpira frekvenčna območja 2.4*GHz* in 5*GHz* ter hitrosti prenosa do 600Mbps. Povečanje hitrost je izvedeno na 3 načine:

1. MIMO tehnologija
   Tehnologija omogoča sočasno uporabo več oddajnih in sprejemnih anten in z tem prenos več podatkovnih tokov (angl. spatial streams) istočasno in neodvisno, odvisno od števila anten.
2. Združevanje kanalov
   Povečanje pasovne širine kanala omogoča prenos več podatkov v istem času, tako npr. kanal širine 40*MHz* omogoča približno dvakrat hitrejši prenos v primerjavi s kanalom širine 20*MHz*. Združiti je mogoče samo dva sosedna neprekrivajoča kanala. V frekvenčnem območju 2.4*GHz* to pomeni, da po združitvi lahko obstaja samo en združeni kanal širine 40*MHz*, zaradi tega se priporoča združevanje le v frekvenčnem območju 5*GHz* kjer obstaja 19 neprekrivajočih kanalov.
3. Združevanje okvirjev
   Režije IEEE 802.11 PHY plasti in MAC podplasti porabijo več časa kot sam prenos podatkov, z združitijo okvirjev pa se poveča prepustnost zaradi združenega prenosa in skupnega potrjevanja okvirjev. Standard definira dve možnosti združevanja okvirjev (slika 2.5):

   – združevanje na protokolu MAC kjer se združijo okvirji z enakimi glavami MAC (združitev MSDU),
   – združevanje na fizični plasti kjer se združijo okvirji protokola MAC (združitev MPDU).

   Slabost združevanja okvirjev na protokolu MAC je ponavljanje celotnega okvirja (vseh MSDU) v primeru neuspešnega prenosa, v nasprotju s PSDU združevanjem, kjer se ponovi prenos samo nepotrjenega okvira MAC.



Slika 2.5: IEEE 802.11n združevanje okvirjev

Frekvenčno območje $5GHz$ je bolj neuporabljeno in posledično ima manj motenj. Zaradi večje pasovne širine omogoča višje hitrosti prenosa. Kot slabost frekvenčnega območja $5GHz$ je pol krajša valovna dolžina in z tem posledično krajši domet omrežja, ker se višje frekvence lažje absorbirajo na ovirah.

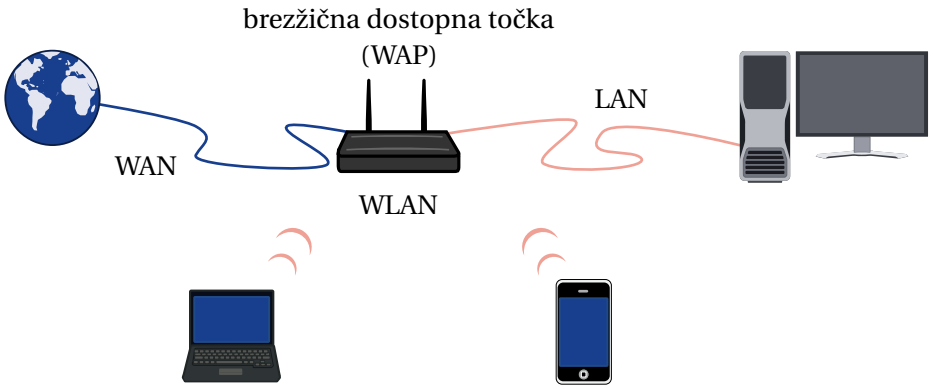
2.5 Vrste povezav brezžičnih postaj

Da bi postaje lahko komunicirale med sabo, oziroma ustvarile omrežje, standard IEEE 802.11 definira dva načina povezav: infrastrukturni in ad hoc.

2.5.1 Infrastrukturni način

Postaje v infrastrukturnem načinu povezave se po vlogi delijo na brezžične dostopne točke (WAP) in odjemalce. Brezžična dostopna točka, običajno brezžični usmerjevalnik, je vgrajeni sistem, ki omogoča povezovanje odjemalcev v infrastrukturni način (BSS). Odjemalci so istočasno lahko povezani z eno dostopno točko, komunikacija med njimi pa poteka le preko dostopne točke. Brezžična omrežja se identificirajo s pomočjo imena SSID. Navadno se uporabljajo kot dopolnitev klasičnega omrežja (slika 2.6). Da se med njima vzpostavi komunikacija, se morata odjemalec in dostopna točka povezati. Povezava poteka v treh korakih:

1. **Iskanje**
   Iskanje je lahko aktivno ali pasivno in je naloga odjemalca. Pri aktivnem iskanju odjemalec pošlje iskalni okvir (angl. probe request frame) na vsakem kanalu, brezžične dostopne točke pa pošljejo okvir z odgovorom (angl. probe response frame), kateri vsebuje ime dostopne točke SSID in lastnosti. Dostopne točke navadno pošiljajo identifikacijske okvirje (angl. beacon frame), običajno vsakih $100ms$, ki vsebujejo ime dostopne točke in lastnosti, odjemalci pa pri pasivnem iskanju iščejo le te. Po zaključku iskanja ima odjemalec informacije o vseh dostopnih točkah.



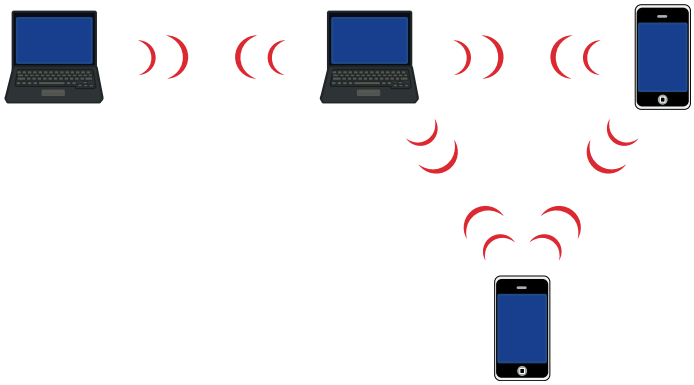
Slika 2.6: Infrastrukturni način (BSS)

2. **Avtentikacija**
   Avtentikacija v infrastrukturnem omrežju je proces preverjanja pravic do uporabe omrežja, izvede pa se pred povezavo v omrežje. Več v poglavju 2.6.

3. **Asociacija**
   Po uspešni avtentikaciji se postaje lahko povežejo, čeprav ni nujno. Nekateri odjemalci uporabljajo predavtentikacijo z vsemi dostopnimi točkami, katerih ključe imajo shranjene, ter si tako zagotovijo hiter prehod med omrežji v primeru izgube signala, dostopne točke pa pomnijo avtentikacije vseh odjemalcev.

2.5.2 Ad hoc način

Ad hoc brezžična povezava je način povezave pri kateri postaje komunicirajo vsaka z vsako – neposredno ali posredno (IBSS) (slika 2.7). Takšne povezave ustvarijo omrežje brez centralne infrastrukture, kjer ima vsaka postaja enakovredno vlogo. Vse postaje sodelujejo pri sprejemanju, pošiljanju in posredovanju okvirjev v odvisnosti od dinamike povezav omrežja. Domet omrežja ni odvisen od prostorske velikosti kolizijske domene brezžične dostopne točke kot pri infrastrukturnem načinu, temveč je odvisen od n–terice povezanih postaj. Postaje pošiljajo identifikacijske okvirje in vzdržujejo informacije o sosednjih postajah, kot so npr. moč signala, interference, zanesljivost, ter na podlagi teh odločajo pri posredovanju okvirjev do končne postaje. Ker okvirji potujejo preko množice medpostaj do končne postaje, je prepustnost takih omrežij bistveno manjša kot pri infrastrukturnih. Običajno se uporablja za kratkotrajna omrežja. Ne podpira avtentikacije uporabnikov, uradno pa podpira samo WEP zaščito.



Slika 2.7: Ad hoc način (IBSS)

2.6 Varnost omrežij

Kot smo že omenili, so okvirji ki jih postaje pošiljajo dostopni vsem postajam v kolizijski domeni. Ker domene ne moremo omejiti kot npr. pri žičnem omrežju, se mora podatke zaščititi, dostop do omrežja pa zagotoviti na osnovi preverljivih avtentikacijskih podatkov.

Kriptografski algoritmi uporabljajo ključ s pomočjo katerega podatek, oziroma črke ali bite podatka, razporedijo na način, da iz njih ni možno določiti pomena podatka (angl. cipher text) brez ključa, obe strani pa morata imeti veljaven ključ. V odvisnosti od časa uporabe, se ključi delijo na dinamične in statične. Ko se isti ključ uporablja pri šifriranju in dešifriranju govorimo o simetričnem ključu, oziroma simetričnem kriptografskem algoritmu. Asimetrična kriptografija temelji na uporabi dveh parov zasebnih in javnih ključev. Če so podatki šifrirani z enim ključem, se lahko dešifrirajo samo s pripadajočim komplementarnim ključem. Taka ključa imenujemo par asimetričnih ključev. Asimetrična kriptografija se zaradi počasnosti običajno uporablja pri avtentikaciji ter izmenjavi simetričnih ključev.

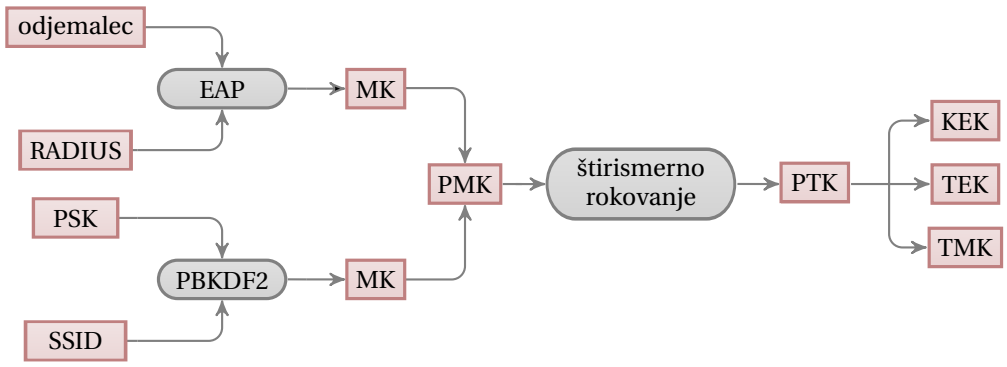
Prvi standard vpeljan leta 1997 je WEP. Uporablja simetričen kriptografski algoritem RC4. Ključ je lahko dolžine 40 ali 104 bitov. Da postaje vedo, da se uporablja WEP zaščita, je enobitno polje WEP v glavi MAC nastavljeno na 1. WEP ponuja dva načina avtentikacije: odprta avtentikacija (angl. open system authentication) in avtentikacija z deljenim ključem (angl. shared key authentication). Odprta avtentikacija je brez zaščitnega mehanizma in dovoljuje vsaki postaji povezavo z dostopno točko. Postopek avtentikacije z deljenim ključem se začne tako, da odjemalec pošlje dostopni točki avtentikacijski okvir (angl. authentication request frame). Dostopna točka odgovori z okvirjem ki vsebuje naključno generiran podatek (angl. authentication respond frame – challenge). Odjemalec ta podatek šifrira z WEP zaščitnim mehanizmom ter pošlje okvir z odgovorom (angl. authentication encryped challenge frame). Dostopna točka dešifrira podatek in ga preveri s poslanim ter v primeru

ujemanja odgovori z okvirom o uspešni avtentikaciji (angl. authentication response frame – success). Avtentikacija tega tipa ni varna zaradi pošiljanja nezaščitenega avtentikaciskega okvirja. Napadalec bi lahko oba okvirja prestregel in s pomočjo orodja za dešifriranje ključev prišel do deljenega ključa. Standard ne podpira avtentikacije uporabnikov.

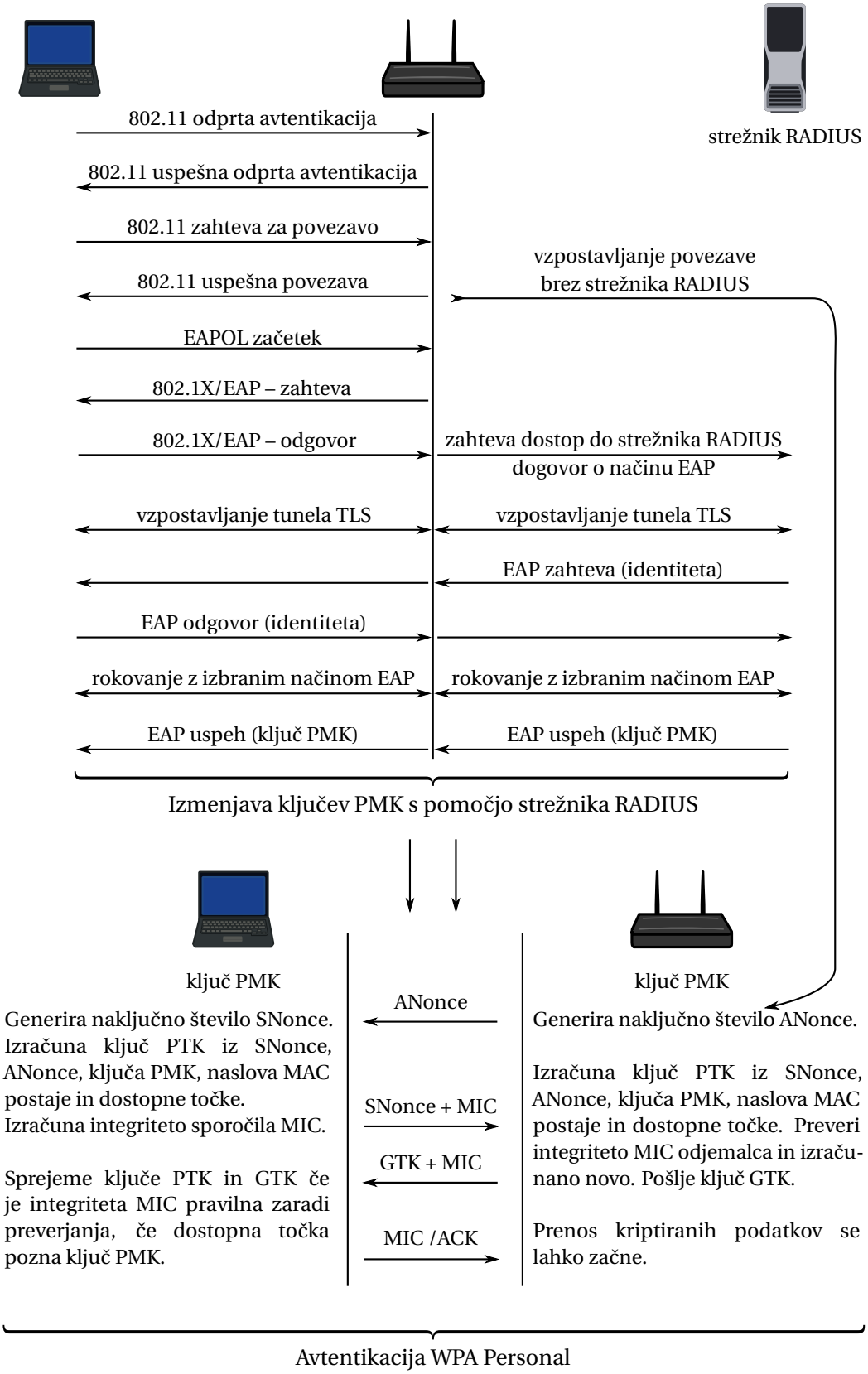
Zaščitni mehanizem WPA implementira večino zaščitnega protokola predpisanega s standardom IEEE 802.11i. Vsak okvir je kriptiran z dinamično generiranim 128 bitnim ključem (TKIP) – tudi z kriptografskim algoritmom RC4. Spremenjeno je tudi preverjanje integritete okvirja, ki je zamenjano z novo zgoščevalno funkcijo MIC.

Zaščitni mehanizem WPA2 je nadgradnja prejšnjega standarda WPA. Razlika med njima je uporaba tehnike spremenljive dolžine ključa (AES). Ključ je lahko dolžine 128, 192 in 256 bitov. Standard ni kompatibilen s starejšimi postajami, saj morajo te strojno podpirati algoritem AES.

WPA in WPA2 omogočata dva načina avtentikacije: WPA Personal in WPA Enterprise (slika 2.9). WPA Personal omogoča avtentikacijo uporabnikov s skupnim geslom (PSK). Namenjen je uporabi v domačih omrežjih ali za omrežja z majhnim številom uporabnikov. WPA Enterprise s pomočjo strežnika RADIUS, s katerim komunicira dostopna točka, preverja in avtenticira uporabnike na podlagi gesla in uporabniškega imena. V postopku avtentikacije se izmenja več ključev z namenom pridobivanja ključa za šifriranje podatkov, vsi pa se pridobijo iz glavnega ključa (MK, slika 2.8). Glavni ključ se pri avtentikaciji WPA Personal izračuna iz gesla in omrežja SSID, pri WPA Enterprise pa se izmenja s protokolom EAP med strežnikom RADIUS, dostopno točko in odjemalcem. Izračun oziroma pridobivanje ključa PMK na obeh straneh je podlaga za postopek pridobivanja ključa PTK s štirismernim rokovanjem z uporabo protokola EAP. Ključ PTK se uporablja za šifriranje podatkov, izmenjavo ključa GTK ter dokazovanje identitete. Ključ GTK se uporablja pri dešifriranju broadcast in multicast prometa, ki se pošilja vsem postajam znotraj omrežja.



Slika 2.8: Generiranje različnih ključev iz glavnega ključa



Slika 2.9: Avtentikacija WPA/WPA2 s strežnikom RADIUS ali brez njega

POGLAVJE 3

Wi-Fi Direct

3.1 Potreba po novem načinu povezovanja postaj

IEEE 802.11, kot je opisano v poglavju 2.5, definira samo infrastrukturni in ad hoc način povezav, oba načina pa imata različne pomanjkljivosti.

Pri infrastrukturnem načinu odjemalci komunicirajo samo z dostopno točko, kar pomeni da se komunikacija med odjemalci odvija posredno, oziroma odjemalec – dostopna točka – odjemalec. Torej se podatki (okvirji) prenašajo dvakrat, odjemalci pa morajo biti člani istega omrežja. Če npr. želimo prenesti večje število podatkov med napravami, npr. datotek slik ali posnetkov, takšen prenos pa ni racionalno opravljati preko medmrežja, moramo napravi, oziroma nekomu zaupati dostop do omrežja, s čimer pa ogrožamo varnost.

Ad hoc način omogoča direktno komunikacijo dveh ali več naprav, povezovanje pa zahteva prekinitev povezave odjemalca in vzpostavljanje ad hoc načina. V tem času izgubimo vse omrežne storitve, ki nam jih infrastrukturna povezava, katere člen smo, ponuja. Tako lahko zgrešimo kakšen klic VOIP, zamudimo pomembno sporočilo, tiskalnik ni dostopen, itn. Sama komunikacija v načinu ad hoc je počasna zaradi topologije omrežja, kjer ne obstaja glavna postaja. Zaradi tega se isti podatki večkrat pošiljajo med napravami z namenom usmerjanja do ciljne postaje. Vse to pa že tako preveč obremenjen brezžični medij še dodatno obremenjuje.

Operacijski sistem Android ima podporo za vzpostavljanje mehke dostopne točke (angl. soft AP) ki omogoča povezovanje odjemalcev v infrastrukturni način. Iz prej omenjenih razlogov je takšna povezava hitrejša in varnejša zaradi sodobnejših varnostnih mehanizmov, ampak se tudi v tem načinu povezave izgubi dostop do infrastrukturnega omrežja.

Torej, glavna pomanjkljivost vseh načinov povezav je nezmožnost postaj, oziroma programske opreme in neobstoja enotnega standarda, da obvladujejo več hkratnih povezav po

standardu IEEE 802.11. Operacijski sistem Linux že dolgo časa brezžično postajo razdeljuje na dva dela, na fizično entiteto PHY in pripadajoče navidezne logične vmesnike. Število fizičnih entitet, kot že ime pove, je enako številu fizično vgrajenih postaj, razen če postaja podpira tehnologijo MIMO. Fizična entiteta omogoča vzpostavljanje različnih navideznih vmesnikov, ki so s strani uporabnika in programske opreme vidni kot fizična postaja. Vsi navidezni vmesniki ene fizične entitete delujejo na istem kanalu. Ukaz `iw list` pokaže vse vmesnike ki jih naprava podpira (koda 3.1).

Programska koda 3.1: Podprti vmesniki brezžične postaje Atheros AR928X

```
Supported interface modes:
Wiphy phy0
         * IBSS
         * managed
         * AP
         * AP/VLAN
         * WDS
         * monitor
         * mesh point
         * P2P-client
         * P2P-GO
```

Nov logični vmesnik v načinu odjemalec se doda z ukazom `iw phy phy0 interface add wlan1 type managed`, ta pa se lahko poveže z drugo dostopno točko. Če v ukazu namesto `managed` podamo `ap`, navidezni vmesnik začne delovati v načinu dostopne točke ter se neposredno z njo lahko povežejo odjemalci. Dodan navidezni vmesnik lahko zagotovi dostop do medmrežja s preslikovalnim protokolom NAT. NAT ustvari preslikovalno tabelo s pomočjo katere usmerja podatke, če pa se ustvari preslikava med navidezno logično dostopno točko in navideznim odjemalcem iste fizične entite, ki je povezan z brezžično dostopno točko, odjemalcem ki so povezani z logično dostopno točko zagotovi dostop do medmrežja. Lahko bi rekli da postaja posreduje, oziroma ponavlja (angl. Wi-Fi repeater) podatke odjemalcev, ki so namenjeni logični dostopni točki do logičnega odjemalca iste postaje, ta pa do fizične brezžične dostopne točke.

Takšen pristop ustvarjanja navideznih ter dinamičnih logičnih vmesnikov je uporabljen pri standardu Wi-Fi Direct, ki je zasnovan pri organizaciji Wi-Fi Alliance, s to razliko, da predpisuje nov logični vmesnik P2P.

3.2 Splošno o standardu Wi-Fi Direct

Standard Wi-Fi Direct [9] predpisuje komponente, topologijo, storive ter naloge storitev pri vzpostavljanju povezav Wi-Fi Direct. Postaje morajo podpirati vsaj standard IEEE 802.11g in varnostni mehanizem WPA2 da bi lahko bile Wi-Fi Direct kompatibilne. Glavna lastnost standarda je, da so postaje med seboj vidne že pred vzpostavljanjem povezave, podobno kot pri ad hoc načinu. Pri vzpostavljanju povezave, med napravami poteka dogovor o vlogah, ki so podobne kot pri infrastrukturnem načinu – vloga dostopne točke in odjemalca. Po uspešnem dogovoru ima ena naprava vlogo dostopne točke, oziroma lastnika skupine, druga pa je odjemalec P2P. Takšen pristop odpravlja vse pomanjkljivosti ad hoc in infrastrukturnega načina.

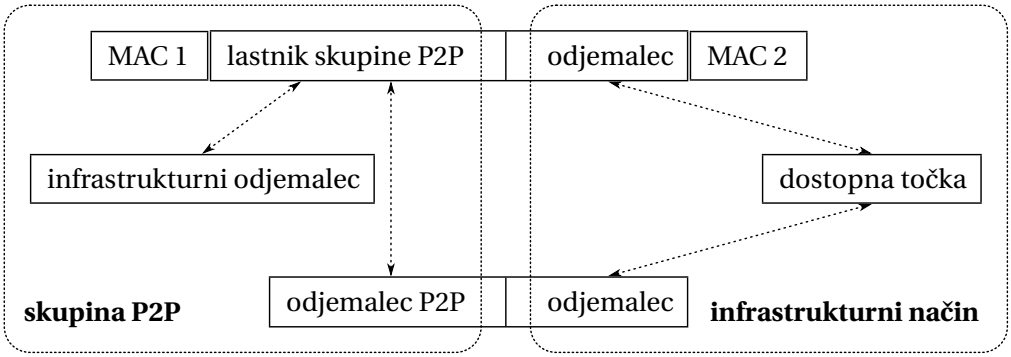
3.3 Arhitektura standarda

3.3.1 Wi-Fi Direct komponente

- Vmesnik P2P ki mora podpirati:
  - vloge lastnika skupine (GO) in odjemalca,
  - dogovarjanje o lastniku skupine,
  - protokol WPS za izmenjavo ključev ter iskanje združljivih naprav,
  - sočasno delovanje kot odjemalec dostopne točke in vmesnik P2P.
- Lastnik skupine:
  - deluje podobno kot dostopna točka,
  - pri protokolu WPS je v vlogi registratorja (angl. WPS registrar),
  - podpira komunikacijo med člani skupine,
  - lahko zagotavlja dostop do dostopne točke članom skupine.
- Odjemalec P2P:
  - deluje podobno kot odjemalec infrastrukturnega načina,
  - pri protokolu WPS je v vlogi odjemalca ključa (angl. WPS enrollee.

3.3.2 Wi-Fi Direct topologija

Topologija 1:n (slika 3.1) kjer je ena ali več P2P odjemalcev povezanih z enim lastnikom skupine se imenuje skupina P2P. Člani skupine so lahko odjemalci P2P in odjemalci infrastrukturnega načina. Skupina P2P se identificira z imenom SSID in predstavlja zasebno



Slika 3.1: Skupina P2P in načini delovanja naprav P2P

varnostno domeno oziroma omrežje. Sočasno delovanje postaje v načinu P2P in načinu infrastrukturnega odjemalca je lahko implementirano na isti fizični entiteti, ali na dveh fizičnih entitetah. Naslov MAC vmesnika P2P in infrastrukturnega odjemalca iste fizične entitete se morata razlikovati.

3.4 Storitve

Dodatne Wi-Fi Direct specifične storitve, ki jih postaja kompatibilna s standardom IEEE 802.11g, zaščitnim mehanizmom WPA2 in protokolom WPS mora podpirati so:

- storitve odkrivanja P2P (angl. P2P discovery),
  - ponujajo nabor funkcij, ki omogočajo identifikacijo naprav in lastnosti le–teh ter povezovanje,
- storitve skupine P2P (angl. P2P group operation),
  - definirajo način delovanja lastnika skupine P2P, ki ima vlogo podobno brezžični dostopni točki,
- varčevanje energije vmesnika P2P (angl. P2P power management),
  - definira nabor funkcij, ki omogočajo zmanjševanje porabe energije vmesnika P2P,
- upravljanje naprav P2P (neobvezno) (angl. managed P2P device operation),
  - opisuje upravljanje naprav P2P v poslovnem okolju, ki jih izvaja oddelek informacijskih tehnologij.

V nadaljevanju se bomo omejili na storitve odkrivanja in storitve skupine.

3.5 Odkrivanje P2P

Odkrivanje P2P omogoča napravam P2P medsebojno vidnost in povezovanje. Razdeljuje se na naslednje storitve:

- odkrivanje naprav P2P (angl. device discovery),
- odkrivanje storitev (angl. service discovery),
  - napravam P2P zagotavlja izmenjavo informacij o storitvah višje plasti pred povezovanjem,
- vzpostavlje skupine (angl. group formation),
- vabilo P2P (angl. P2P invitation),
  - uporablja se za vzpostavljanje obstojne skupine P2P (angl. persistent P2P group) ali povabitev naprave P2P v že vzpostavljeno skupino.

3.5.1 Odkrivanje naprav P2P

Medsebojno odkrivanje lastnosti naprav P2P, podobno kot pri infrastrukturnem načinu pri iskanju brezžičnih dostopnih točk, je sestavljeno iz iskalnih okvirjev (angl. probe request frame) in okvirjev z odgovorom (angl. probe response frame). Naprava P2P pošilja okvirje z odgovorom samo če je v naslednjih stanjih oziroma vlogah:

- je lastnik skupine,
- je v stanju poslušanja (angl. listen state),
- če je povezana z dostopno točko na istem kanalu kot poslan iskalni okvir in če ni član skupine.

Identifikacijske okvirje (angl. beacon frame), podobno kot dostopna točka, pošilja samo v primeru, ko je lastnik skupine. Odkrivanje naprav se deli na tri stanja oziroma faze:

1. Stanje poslušanja (angl. listen state)
   V stanju poslušanja naprava P2P posluša iskalne okvirje P2P na prej izbranem kanalu. V tem stanju odgovarja na iskalne okvirje, na okvirje za vzpostavitev skupine ter zahtevami za povezovanje.
2. Faza iskanja (angl. scan phase)
   Naprava P2P pošilja iskalne okvirje na ti. družabnih kanalih (angl. social channels), ki

so 1, 6 in 11 v frekvenčnem območju 2.4*GHz*. Iskalni okvirji so sestavljeni iz P2P informacijskega elementa (50-6F-9A, koda 3.2, vrstica 18), SSID P2P, ki se začne z DIRECT-, naslovov cilja in sprejemnika ki so v formatu FF:FF:FF:FF:FF:FF, kar pomeni da so naslovljene vse naprave, naslovov oddajnika in izvora (vrstice 7,8) ter polja z informacijami WPS.

3. Faza ugotavljanja (angl. find phase)
   Namen faze ugotavljanja je zagotavljanje, da se dve napravi, ki sta v stanju iskanja in poslušanja najdeta oziroma postavita na skupni kanal zaradi zagotovitve komunikacije.

Programska koda 3.2: Iskalni okvir P2P zajet z aplikacijo Wireshark

```
1  IEEE 802.11 Probe Request, Flags: ........C
2      Type/Subtype: Probe Request (0x04)
3      Frame Control Field: 0x4000
4      .000 0000 0000 0000 = Duration: 0 microseconds
5      Receiver address: Broadcast (ff:ff:ff:ff:ff:ff)
6      Destination address: Broadcast (ff:ff:ff:ff:ff:ff)
7      Transmitter address: 02:08:22:72:46:fc
8      Source address: 02:08:22:72:46:fc
9      BSS Id: Broadcast (ff:ff:ff:ff:ff:ff)
10     Fragment number: 0
11     Sequence number: 512
12     Frame check sequence: 0xedd78894 [correct]
13 IEEE 802.11 wireless LAN management frame
14     Tagged parameters (189 bytes)
15         Tag: SSID parameter set: DIRECT-
16         Tag: Supported Rates 6, 9, 12, 18, 24, 36, 48, 54, [Mbps]
17         Tag: Vendor Specific: Microsof: WPS
18         Tag: Vendor Specific: Wi-FiAll: P2P
```

Ko se naprava P2P želi povezati z napravo, ki je že član skupine, oziroma ta v tem primeru ne more odgovarjati na iskalne okvirje, lastnik skupine namesto nje odgovori. Odgovor (koda 3.3) je sestavljen iz imena skupine SSID (vrstica 10), informacij o lastniku skupine (vrstica 19), informacij o skupini (vrstica 32) in članov skupine (vrstica 53). Naprava P2P ima več možnosti vzpostavljanja povezave z napravo ki je že član skupine:

- pridruži se skupini ciljne naprave,
- pošlje vabilo ciljni napravi da se pridruži skupini katere je član ali lastnik,
- pošlje vabilo za vzpostavljanje prejšnje obstojne skupine,
- poskusi vzpostavljanje nove skupine s ciljno napravo, če je ta naprava zmožna biti član še katere skupine (vrstica 43).

Programska koda 3.3: Okvir z odgovorom P2P zajet z aplikacijo Wireshark

```
IEEE 802.11 Probe Response, Flags: ........C
    Receiver address: 00:22:43:51:67:17
    Destination address: 00:22:43:51:67:17
    Transmitter address: 02:1d:c9:91:18:3f
    Source address: 02:1d:c9:91:18:3f
    BSS Id: 02:1d:c9:91:18:3f
IEEE 802.11 wireless LAN management frame
    Fixed parameters (12 bytes)
    Tagged parameters (341 bytes)
        Tag: SSID parameter set: DIRECT-MQ
        Tag: DS Parameter set: Current Channel: 6
        Tag: Vendor Specific: Microsof: WPS
        Tag: Vendor Specific: Wi-FiAll: P2P
            Tag Number: Vendor Specific (221)
            Tag length: 88
            OUI: 50-6f-9a (Wi-FiAll)
            Vendor Specific OUI Type: 9
            P2P Capability: Device 0x33  Group 0x49
            P2P Device Info
                Attribute Type: P2P Device Info (13)
                Attribute Length: 33
                P2P Device address: 02:1d:c9:91:18:3f
                Config Methods: 0x430c
                Primary Device Type: 00060050f2040001
                Primary Device Type: Category: 6
                Primary Device Type: OUI: 0050f204
                Primary Device Type: Subcategory: 1
                Number of Secondary Device Types: 0
                Device Name attribute type: 0x1011
                Device Name attribute length: 12
                Device Name: GAINSPAN-P2P
            P2P Group Info
                Attribute Type: P2P Group Info (14)
                Attribute Length: 40
                P2P Client Info Descriptor Length: 39
                P2P Device address: 02:08:22:72:46:fc
                P2P Interface address: 02:08:22:72:46:fc
                Device Capability Bitmap: 0x27
                .... ...1 = Service Discovery: 0x01
                .... ..1. = P2P Client Discoverability: 0x01
                .... .1.. = Concurrent Operation: 0x01
                .... 0... = P2P Infrastructure Managed: 0x00
                ...0 .... = P2P Device Limit: 0x00
                ..1. .... = P2P Invitation Procedure: 0x01
                Config Methods: 0x0188
                Primary Device Type: 000a0050f2040005
                Primary Device Type: Category: 10
                Primary Device Type: OUI: 0050f204
                Primary Device Type: Subcategory: 5
                Number of Secondary Device Types: 0
                Device Name attribute type: 0x1011
                Device Name attribute length: 13
                Device Name: Prestigio4500
```

3.6 Vzpostavljanje skupine P2P

3.6.1 Dogovor o lastniku skupine

Dogovor poteka z izmenjavo treh okvirjev med dvemi napravami. S prvim (ang. GO negotiation request) se začne dogovor o skupini. Naprava, ki zahteva ustvarjanje skupine pošlje v okvirju število z namenom o lastništvu skupine (angl. GO intent), ki je velikosti od 0 do 15. Druga naprava odgovori z okvirjem (ang. GO negotiation response), ki tudi vsebuje število z namenom o lastništvu skupine. Na koncu naprava, ki je zahtevala ustvarjanje skupine pošlje odgovor da je sprejela okvir (angl. GO negotiation confirmation). Po izmenjavi okvirjev imata obe napravi števila z namenom o lastništvu skupine (koda 3.4). Naprava ki ima večje število ustvari skupino, katere lastnik je. Če sta števili enaki (angl. tie breaker), naprava ki ima nastavljen bit (vrstica 5) ustvari skupino.

Programska koda 3.4: Del okvirja pri vzpostavljanju skupine

```
1 Group Owner Intent: Intent 15  Tie breaker 0
2    Attribute Type: Group Owner Intent (4)
3    Attribute Length: 1
4    ...1 111. = Group Owner Intent: 15
5    .... ...0 = Group Owner Intent Tie Breaker: 0
```

3.6.2 Avtonomna skupina

Avtonomna skupina se ustvari brez dogovora o lastniku skupine. Naprava P2P, ki želi biti lastnik skupine jo samostojno ustvari. Naprave, ki se želijo pridružiti skupini, se pridružijo na enak način, kot da bi bila skupina ustvarjena z dogovorom z drugo napravo.

3.6.3 Obstojna skupina

Pri vzpostavljanju skupine lahko naprave določijo, da je skupina obstojna z identifikacijskimi in iskalnimi okvirji, po vzpostavljanju skupine pa shranijo ključe in vloge. Pri odkrivanju naprave P2P spoznajo, da so v preteklosti vzpostavile skupino in s pomočjo vabila ponovno ustvarijo skupino. Vzpostavljanje skupine je hitrejše, ker poteka brez dogovora o lastniku in brez izmenjave ključa s pomočjo protokola WPS.

3.7 Avtentikacija

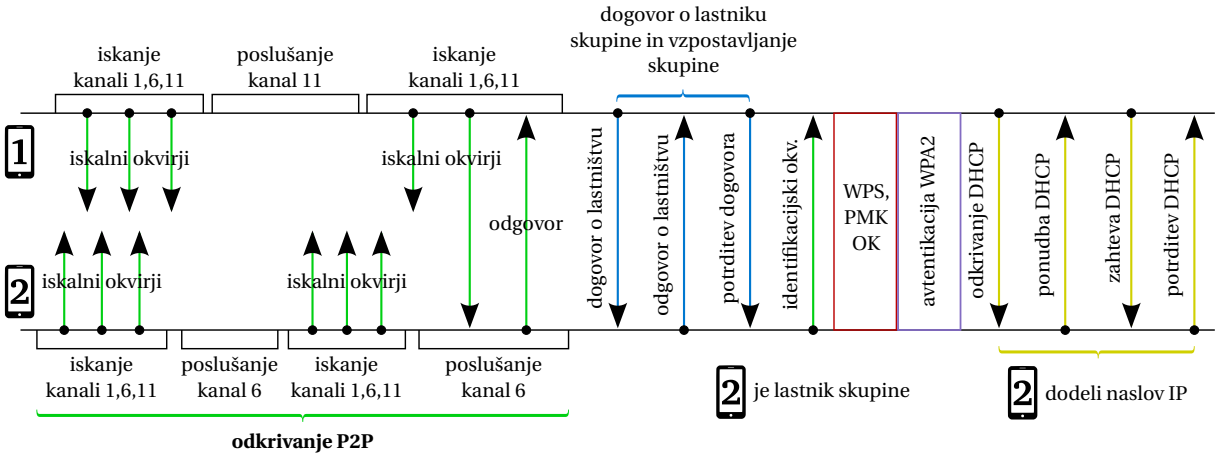
Po vzpostavitvi skupine, je napravam, ki niso Wi-Fi Direct združljive, skupina vidna kot navadna brezžična dostopna točka. Wi-Fi Direct zaradi enostavnejšega povezovanja za naprave P2P predpisuje izmenjavo ključa s protokolom WPS. Naprave P2P se po vlogi v protokolu WPS delijo na registratorja (angl. registrar), ki ponuja ključ in je lastnik skupine in naprave P2P, ki zahteva ključ (angl. enrollee), oziroma se pridružuje skupini. Protokol omogoča izmenjavo ključa s pomočjo več različnih metod (koda 3.5), prenos ključa med napravami pa poteka s protokolom EAP.

Programska koda 3.5: Metode WPS zajete pri standardu Wi-Fi Direct

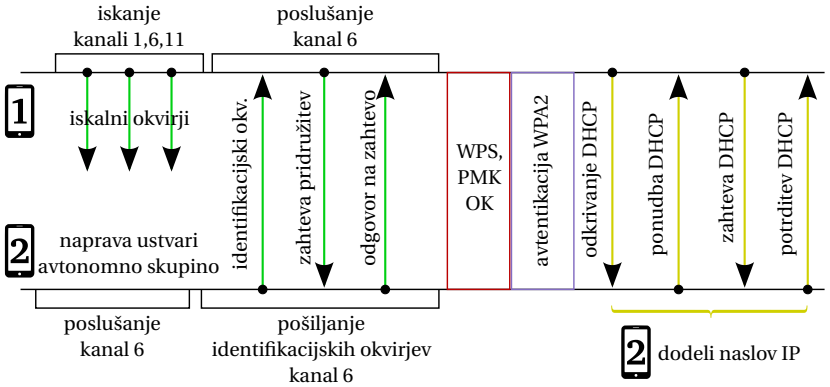
```
    Tag: Vendor Specific: Microsof: WPS
        Config Methods: 0x430c
            Data Element Type: Config Methods (0x1008)
            Data Element Length: 2
            Configuration Methods: 0x430c
            .... .... .... ...0 = USB: 0x0000
            .... .... .... ..0. = Ethernet: 0x0000
            .... .... .... .1.. = Label: 0x0001
            .... .... .... 1... = Display: 0x0001
            ..0. .... .... .... = Virtual Display: 0x0000
            .1.. .... .... .... = Physical Display: 0x0001
            .... .... ...0 .... = External NFC: 0x0000
            .... .... ..0. .... = Internal NFC: 0x0000
            .... .... .0.. .... = NFC Interface: 0x0000
            .... .... 0... .... = Push Button: 0x0000
            .... ..1. .... .... = Virtual Push Button: 0x0001
            .... .0.. .... .... = Physical Push Button: 0x0000
            .... ...1 .... .... = Keypad: 0x0001
```

Najenostavnejša metoda je povezovanje s pomočjo tipke (angl. push button), ki je lahko programsko ali fizično implementirana. Naprava P2P, ki zahteva povezavo čaka, da uporabnik na drugi strani pritisne tipko, s čimer potrdi, da se strinja s povezovanjem. Po pritisku tipke se ključi PMK izmenjajo, zatem pa se lahko začne proces avtentikacije, enako kot pri WPA Personal. Izmenjava ključa z metodo PIN zahteva od uporabnika da vpiše (keypad) PIN, ki ga druga naprava prikaže (display), naprave pa morajo podpirati komplementarnost PIN metode (keypad – display). PIN je sestavljen iz 8 številk, zadnja številka je kontrolna in je izračunana iz ostalih. NFC način ponuja izmenjavo ključev tako, da se napravi približajo v domet NFC sprejemnikov.

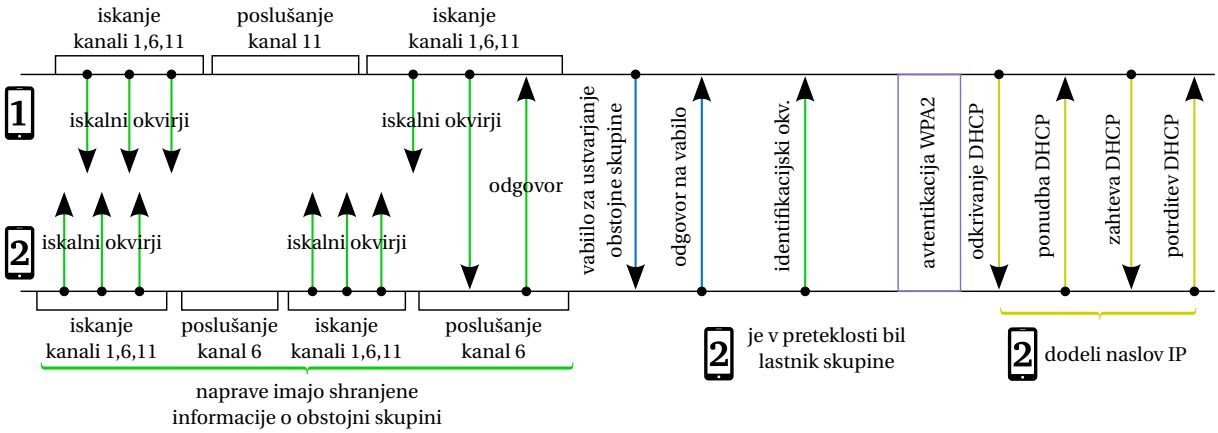
Po avtentikaciji sledi pridruževanje člana skupini. Da bi naprave lahko medsebojno komunicirale na višjih plasteh, oziroma s protokoli TCP/IP ali UDP, se morajo napravam dodeliti naslovi IP. Standard predpisuje, da na napravi P2P ki je lastnik skupine teče strežnik DHCP. Podroben potek vzpostavljanja skupine na vse tri načine prikazuje slika 3.2.



(a) Vzpostavljanje skupine z dogovorom o lastniku



(b) Vzpostavljanje avtonomne skupine



(c) Vzpostavljanje obstojne skupine

Slika 3.2: Vzpostavljanje skupine P2P na vse tri načine

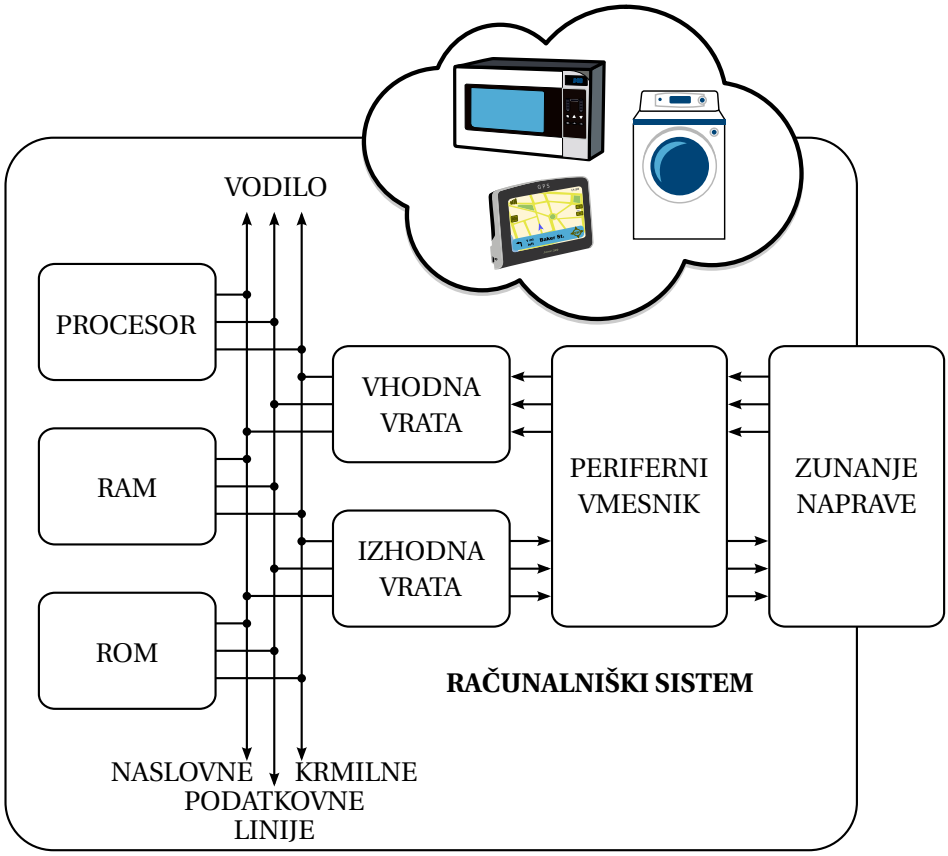
POGLAVJE 4

Mikrokrmilna knjižnica Arduino za GainSpan 1500M

4.1 Vgrajeni sistemi

Obstaja več različnih definicij vgrajenih sistemov (slika 4.1), najbolj tipična pa je, da je to namenski računalniški sistem, ki je del večjega sistema, stroja ali naprave. Pomen izraza lahko razčlenimo na besedo vgrajen, kar pomeni, da je skrit v notranjosti, ter sistem, ki nakazuje, da gre za skupek komponent, ki delujo za dosego cilja – so ključne za upravljanje, nadzor in krmiljenje delovanja naprave, ki bi jo lahko po tem ključu tudi imenovali vgrajena naprava. Ker imajo takšne naprave omejene in natančno določene funkcionalnosti (npr. GPS navigacija, predvajalnik mp3), morata biti programska oprema in računalniški sistem prilagojeni namenu naprave ter optimizirani s strani stroškov izdelave, zmogljivosti, zanesljivosti in energijske učinkovitosti. Vgrajen sistem ima enake osnovne gradnike kot vsak računalniški sistem – mikroprocesor, podatkovni in programski pomnilnik, vodilo ter periferne naprave. Mikroračunalnik je dimenzijsko manjši računalniški sistem sestavljen iz osnovnih gradnikov, če so pa le ti integrirani na eni silicijevi rezini, dobimo mikrokrmilnik ali mikrokontroler. Kot glavne lastnosti vgrajenih sistemov lahko izpostavimo:

- vsebujejo program, ki se avtomatično naloži ob zagonu sistema,
- zagotavljajo nove in naprednejše funkcionalnosti in inteligenco,
- izvajajo samo nabor bolj ali manj specifičnih ter kompleksnih opravil,
- do določene meje delujejo samostojno.



Slika 4.1: Osnovne komponente vgrajenega sistema

4.1.1 Mikrokrmilnik

V splošnem je mikrokrmilnik računalniški sistem v malem, ki v enem integriranem vezju združuje mikroprocesor, pomnilnik (RAM, ROM), vodilo in vhodno izhodne periferne vmesnike. Namenjen je krmiljenju in zaznavanju zunanjih (perifernih) naprav, njegovo funkcionalnost pa definirajo program in zunanje naprave. Standardni vhodi splošnonamenskih računalnikov so miška in tipkovnica, izhodi pa zaslon in tiskalnik. Pri mikrokrmilnikih pa je situacija bistveno drugačna, namreč ti imajo celo paleto najosnovnejših perifernih vmesnikov že integriranih, ki so programsko enostavno dostopni in nastavljivi. Najbolj pogosti so pretvorniki AD in DA, modulatorji PWM in serijski vmesniki.

Naloga pretvornika AD, je da analogno vrednost na vhodu mikrokrmilnika spremeni v digitalno. Običajno se uporablja za merjenje vhodne napetosti ali posredno upora komponente, ki se v določenih pogojih spreminja. To so npr. senzorji toplote, vlažnosti, potenciometri itn. Pretvornik AD vrednost iz zveznega analognega signala s pomočjo vzorčevalno–zadrževalnega vezja v enakomernih časovnih intervalih odtipa in jo spremeni v diskretno digitalno vrednost, ki je primerna za obdelavo s programom mikrokrmilnika.

Pretvornik DA dela obratno od pretvornika AD. V programu definirano digitalno vrednost spremeni v izhodno analogno, npr. vrednost HI (angl. visoko) spremeni v maksimalno dovoljeno vrednost napetosti na izbranem izhodu mikrokrmilnika. Takšna vrednost lahko poganja npr. LED, piezo zvočnik, z vmesnimi prilagoditvenimi elektronskimi elementi pa lahko krmili tudi močnejše naprave.

Vmesnik PWM omogoča generiranje pravokotnega signala s pomočjo impulzov med napetostjo nič in napajalno napetostjo. Frekvenca in dolžina impulza določata analogno vrednost napetosti, ki jih naprave zaradi slabe odzivnosti vidijo kot zvezno. Kot primer lahko vzamemo že izginjajoče žarnice z žarilno nitjo. Ko so priklopljene na sinusno frekvenco $50Hz$, je napetost na žarilni niti vsakih $0,02s$ enaka $0V$, ker pa ima dovolj vztrajnosti, da ne ugasne, jo človeško oko zazna kot neprekinjeno svetlobo.

Serijski vmesnik je elektronsko vezje ki podpira serijski prenos podatkov. Namesto da se vsak bit pošilja po svoji liniji, kot pri vzporednemu, se biti pošiljajo v zaporedju impulzov po eni liniji. Glede na prenos urinega signala obstaja sinhrona (prenaša se tudi urin signal) in asinhrona (prenašajo se samo podatki) komunikacija. V odvisnosti od števila prenosnih linij razlikujemo:

– simplex – prenos v eno smer (ena linija),

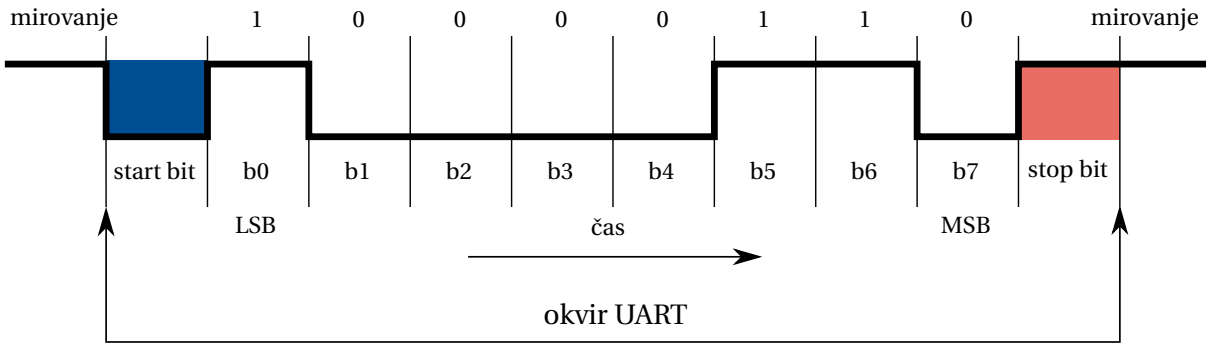
– half-duplex – dvosmerna komunikacija, a le v eno smer naenkrat (ena linija),

– full-duplex – dvosmerna komunikacija v obe smeri naenkrat (dve liniji).

Obstaja več različic serijskih protokolov, kot so npr. UART, RS–232, SPI, I2C. Pri UART popolnoma dvosmerni (full-duplex) komunikaciji so linije običajno označene z RX(D) in TX(D). Pri povezovanju z drugo napravo se povežejo navzkrižno (RX–TX, TX–RX). Povezane naprave morajo komunicirati z enako hitrostjo (bps, baud), z enakim številom naenkrat prenašanih bitov ter z enako napetostno predstavitvijo posameznega bita. Startni bit (slika 4.2) je nizko stanje in označuje začetek prenosa, konec je stop bit in je visoko stanje. V času mirovanja je linija v visoki napetosti. Razlaga izhaja že iz časov analognih telekomunikacij zaradi enostavnega preverjanja če je linija prekinjena. Logična vrednost 1 je predstavljena z visoko napetostjo, logična vrednost 0 pa z $0V$.

4.1.2 Wi-Fi modul GainSpan 1500M

Wi-Fi modul GainSpan 1500M je vgrajen sistem, ki z osnovno programsko opremo podpira vse vloge, oziroma načine komunikacij definirane s standardom IEEE 802.11. Podprte so različice standarda IEEE 802.11 b/g/n hitrosti do 72.2Mbps ter vse različice brezžičnih varnostnih



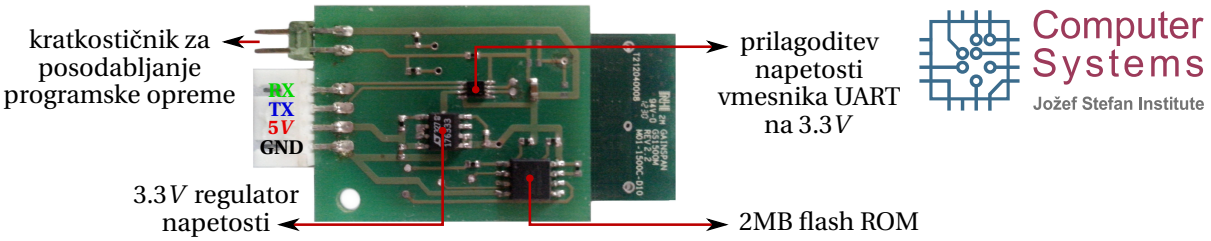
Slika 4.2: Prenos črke a (01100001_2)

mehanizmov. Na višjih plasteh referenčnega modela ISO/OSI podpira 15 TCP/IP in UDP povezav ki so lahko v načinu strežnik in odjemalec, strežnik in odjemalec DHCP ter protokole HTTP, DNS, ICMP in SSL.

Za komunikacijo z zunanjimi napravami sta zadolžena dva vmesnika UART hitrosti do 921.6kbps in dva vmesnika SPI hitrosti do 3Mbps. Dva mikroprocesorja ARM7 sta zadolžena za omrežne storitve in krmiljenje brezžičnega sprejemnika, delijo pa si 128KB pomnilnika RAM. Pomnilnik ROM velikosti 384KB je namenjen shranjevanju programske opreme, razdeljen pa je na dva dela – 128KB je namenjeno za programsko kodo, ki upravlja z brezžičnim sprejemnikom, 256KB pa za storitve višjih plasti referenčnega modela ISO/OSI. Za posodabljanje programske kode in upravljanje s serijskimi vmesniki je zadolžen zaganjalnik, ki je shranjen v dodatnem pomnilniku ROM velikosti 80KB, aktivira pa se z napetostjo $3.3\,V$ na nožici GPIO27 pred zagonom modula.

Tiskano vezje za Wi-Fi modul GainSpan 1500M razvito na Institutu "Jožef Stefan", – Odsek za računalniške sisteme (slika 4.3) ima več nalog:

1. prilagajanje vhodne napajalne napetosti na $3.3\,V$,
2. prilagoditev vhodnih nivojev vmesnika UART na $3.3\,V$,
3. modul z osnovno programsko opremo ne podpira standarda Wi-Fi Direct zaradi nezadostne velikosti vgrajenega flash pomnilnika, zaradi tega je potrebno dodatni zunanji flash pomnilnik povezati z modulom,
4. vgrajen kratkostičnik omogoča dovajanje napetosti $3.3\,V$ na nožico GPIO27 in posodabljanje programske opreme.



Slika 4.3: Tiskano vezje modula GainSpan 1500M razvito na Institutu "Jožef Stefan"

4.1.3 Mikrokrmilna platforma Arduino

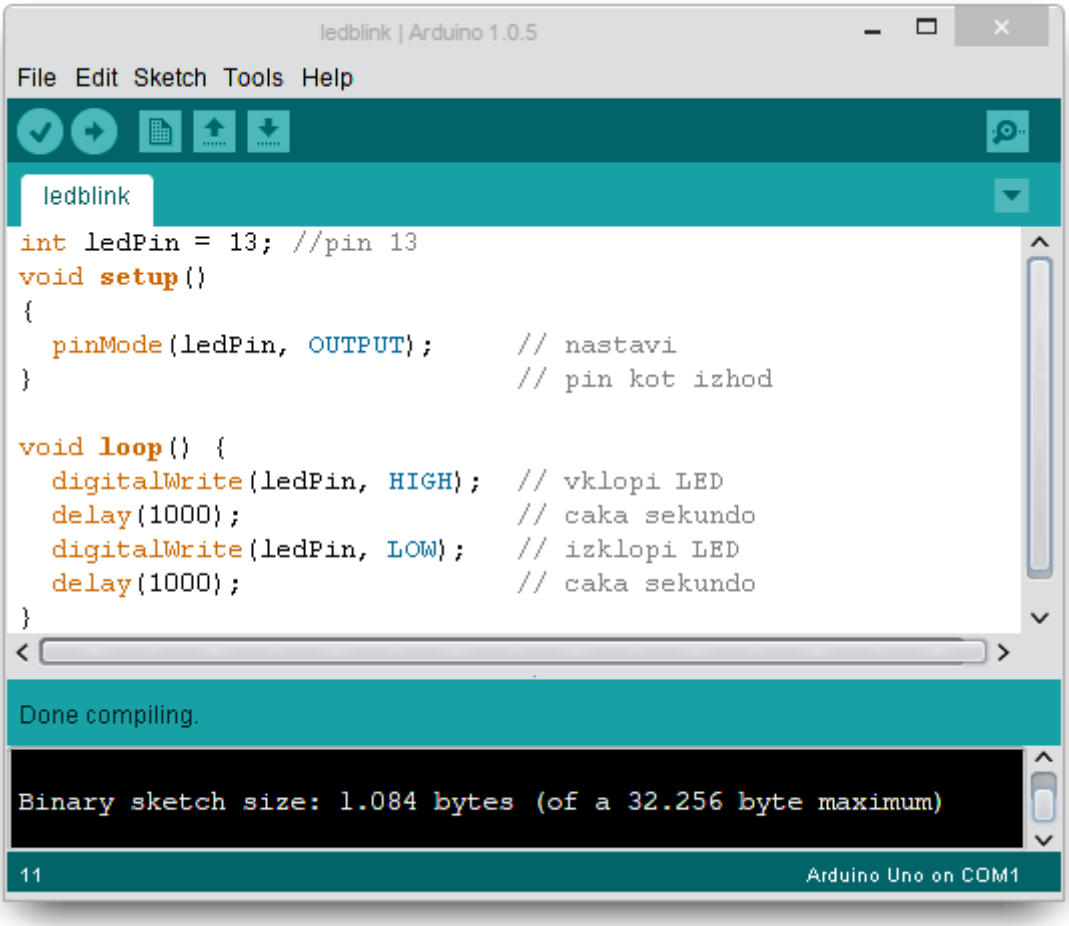
Mikrokrmilna platforma Arduino je celovit koncept odprtokodne programske in strojne opreme, ki temelji na Atmelovem mikrokrmilniku iz družine AVR.

Programska oprema

Programska oprema je sestavljena iz razvojnega okolja Arduino IDE (slika 4.4) in programskega jezika Arduino. Programski jezik Arduino je nastal na osnovi odprtokodnih razvojnih orodij za programiranje mikrokrmilnikov Wiring [1], osnova za razvojno okolje pa je Processing [2]. Razvojno okolje omogoča pisanje programske kode, prevajanje in zapisovanje uspešno prevedene kode v mikrokrmilnik. Programska koda višjega programskega jezika Arduino, ki je podobna klasičnemu jeziku C, se prevaja z odprtokodnim prevajalnikom GCC. V primeru Arduina, je GCC navzkrižni prevajalnik (angl. cross compiler), kar pomeni da se programska koda, ki jo prevaja ne bo izvajala na računalniški arhitekturi, kjer se prevajanje izvaja, ampak na drugi, v primeru Arduina na AVR arhitekturi mikroprocesorja. Slika 4.4 prikazuje tudi osnovno zgradbo vsakega Arduino programa, funkciji `setup()` in `loop()`. Funkcija `setup()` se izvede na začetku programa, njena naloga pa je inicializacija nastavitev. Funkcija `loop()` izvaja glavni del programa in se izvaja do izklopa mikrokrmilnika. Razvojno okolje ima že vgrajene knjižnice za podporo strojne opreme Arduino, knjižnice za podporo proizvodov drugih proizvajalcev pa se uvozijo s pomočjo orodne vrstice `Sketch→Import library`. Vgrajen prikazovalnik sporočil UART omogoča sprejemanje in pošiljanje sporočil (ukazov) mikrokrmilnika. Je dokaj okrnjen, ne podpira pošiljanja in sprejemanja sporočil v šesnajstiški obliki, pošiljanja datotek ter pomnjenja zadnjih ukazov. Iz teh razlogov smo raje uporabljali odprtokodno aplikacijo CuteCom.

Strojna oprema – Arduino UNO

Arduino UNO je prototipna platforma, ki temelji na Atmelovem mikrokrmilniku ATmega328. Ima vgrajene vse pomožne elemente, ki jih mikrokrmilnik potrebuje za samostojno delo.



Slika 4.4: Razvojno okolje Arduino

Primerna je predvsem za prototipni razvoj vgrajenega sistema, ker ima vse priključke povezane in označene. Zasnovana je modularno, kar omogoča dodatne funkcionalnosti s pomočjo ščitov (angl. shields). Ščiti se enostavno priklapljajo na platformo, lahko pa se jih doda tudi več v odvisnosti od namena in vrste komunikacijskega protokola. Če ščiti komunicirajo s serijskim protokolom SPI, ki podpira več naprav na istem vodilu, potem jih lahko fizično dodamo več ter sočasno krmilimo. Arduino proizvaja različne ščite kot so npr. GSM, WiFi, Ethernet, SD itn. Zaradi odprtosti platforme obstaja veliko število kompatibilnih ščitov drugih proizvajalcev. Dodaten mikrokrmilnik Atmega16U2 omogoča prenos podatkov serijskega protokola UART preko vmesnika USB.

Vgrajen vmesnik USB ima več nalog:

- napaja prototipno platformo,
- prenaša sporočila UART v obe smeri,
- omogoča enostavno nalaganje programske kode.

Lastnosti ATmega328 mikrokrmilnika:

- 8 bitni procesor RISC arhitekture hitrosti 16*MHz*,
- 2KB SRAM in 32KB flash ROM pomnilnika,
- delovna napetost 5*V*,
- 6 analognih vhodov in 14 digitalnih (od tega jih 6 lahko deluje v načinu PWM),
- 1 full-duplex strojni vmesnik UART.

Mikrokrmilnik Arduino UNO platforme ima prednaložen zaganjalnik (angl. bootloader). Zaganjalnik je prvi program, ki se zažene po resetu ali vklopu mikrokrmilnika, njegova naloga pa je, da odgovori na zahtevo po nalaganju programske kode iz Arduino razvojnega okolja – sicer zažene že shranjen program.

4.2 Potreba po mikrokrmilni knjižnici

Wi-Fi modul GainSpan 1500M pri komunikaciji s standardom IEEE 802.11 podpira shranjevanje nastavitev brezžične komunikacije, kot so način, ime omrežja katerega vzpostavlja ali s katerim se povezuje, vrsto zaščitnega mehanizma in ključe. Na višjih plasteh modela ISO/OSI podpira shranjevanje naslovov IP, ki jih lahko statično nastavi, če pa modul deluje v načinu dostopne točke, strežnik DHCP iz shranjenih nastavitev IP dodeljuje naslove IP v rangu omrežja in velikosti podomrežja. Na višji plasti referenčnega modela ISO/OSI, kjer poteka komunikacija s protokoli TCP/IP in UDP omogoča shranjevanje naslovov IP in vrat strežnika in odjemalca. Podpira tudi shranjevanje lastnosti serijskega vmesnika UART kot so hitrost, število naenkrat poslanih bitov ter nastavitve parnosti.

Avtomatično vzpostavljanje brezžične in TCP/IP ali UDP povezave iz shranjenih nastavitev v pomnilniku modula se po vklopu ali po shranjevanju zažene z ukazom `ATA`, ukaze pa sprejema preko vmesnika UART. Če želimo vzpostaviti samo povezavo TCP/IP ali UDP pa uporabimo ukaz `ATA2`. Ukaz `ATA` se lahko avtomatično izvede vsakič po vklopu modula z ukazom `ATC1`. Modul v tem načinu podpira samo eno povezavo TCP/IP ali UDP, torej naenkrat lahko komunicira samo z enim odjemalcem ali strežnikom. Sporočila TCP/IP ali UDP se izpisujejo direktno preko vmesnika UART, brez kakršnihkoli informacij od katerega strežnika, oziroma odjemalca so prišle. Pošiljanje sporočil je podobno, direktno se vpisujejo na vmesnik UART ter pošiljajo odjemalcu ali strežniku s katerim je vzpostavljena povezava.

Avtomatičnega vzpostavljanja povezave Wi-Fi Direct modul ne podpira. Pri tem mislimo na to, da bi shranili informacije povezave Wi-Fi Direct, kot so ime naprave, način WPS, število PIN ter število o lastniku skupine. Po inicializaciji teh nastavitev bi modul čakal zahtevo za

povezovanjem druge naprave, podobno kot pri dostopni točki, in če ima ta naprava enak PIN, se povezava, oziroma skupina samodejno vzpostavi. Ker modul takšne inicializacije ne podpira, se mora vzpostavljanje povezav Wi-Fi Direct in TCP/IP ali UDP krmiliti preko vmesnika UART.

Zaradi pomanjkljivosti avtomatičnega vzpostavljanja povezav Wi-Fi Direct je bilo potrebno napisati mikrokrmilni program, v našem primeru knjižnico za mikrokrmilno platformo Arduino UNO. Knjižnica upravlja z vzpostavljanjem povezave tako, da modulu pošilja primerne ukaze preko vmesnika UART mikrokrmilnika. Z branjem sporočil modula se preverja pravilnost odziva modula, na podlagi odziva in stanja modula pa se odloča o naslednjem ukazu modulu. S povezavo omenjenih vgrajenih sistemov ter z uporabo mikrokrmilne knjižice smo dosegli avtomatično krmiljenje vzpostavljanja povezav Wi-Fi Direct.

4.3 Testno vzpostavljanje povezav Wi-Fi Direct

Preden smo lahko začeli s programiranjem mikrokrmilne knjižnice, smo morali dobiti občutek kako se v operacijskemu sistemu Linux, oziroma Android, vzpostavlja povezava Wi-Fi Direct. Testiranje smo izvajali med pametnim telefonom Prestigio 4500 katerega poganja Android različica 4.1.1 in prenosnikom z operacijskim sistemom Gentoo Linux z vgrajeno brezžično kartico Atheros AR928X b/g/n.

Obe napravi za vzpostavljanje povezav Wi-Fi Direct uporabljata isto sistemsko storitev `wpa_supplicant` [4]. Dobra stran storitve je da podpira ukazni način s pomočjo programskega orodja `wpa_cli`. Tako lahko natančno vidimo postopek in korake vzpostavljanja povezave Wi-Fi Direct. Testiranje povezljivosti med napravami, ki uporabljajo enako storitev za vzpostavljanje povezav Wi-Fi Direct je zaradi tega bolj predvidljivo in ponovljivo, odprtokodnost storitve ter veliko prostodostopne dokumentacije pa nam pomaga pri razumevanju nejasnosti specifikacij standarda.

Storitev `wpa_supplicant` bere osnovne nastavitve iz datoteke, ki jo podamo kot parameter ukazne vrstice (koda 4.1). Zagon in upravljanje s storitvijo `wpa_supplicant` je na računalniku sestavljeno iz dveh korakov:

1. zagon in inicializacija `wpa_supplicant` nastavitev
   `wpa_supplicant -d -Dnl80211 -cwpa_supplicant.conf -iwlan0 -B`
2. zagon ukaznega programa `wpa_cli` na računalniku
   `wpa_cli`

Programska koda 4.1: Osnovne nastavitve storitve `wpa_supplicant`

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=wheel
update_config=1
device_name=PC-P2P   #ime naprave vidno pri iskanju
p2p_go_intent=0      #število o lastniku skupine
device_type=1-0050F204-1
p2p_go_ht40=1
```

Ko smo v `wpa_cli` ukaznem načinu lahko spreminjamo lastnosti povezave oziroma naprave, kot so npr. število o lastništvu skupine in izbira metod WPS ter podajamo ukaze iskanja, povezovanja, vnos PIN itn.

Operacijski sistem Android v sistemskem grafičnem vmesniku za upravljanje povezav Wi-Fi Direct ne dopušča izbire metod WPS in nastavitve števila o lastniku skupine, temveč so lastnosti povezave odvisne od podprtih lastnosti naprave s katero se vzpostavlja povezava.

4.3.1 Vzpostavljanje povezave z metodo WPS – tipka

Naprava Android zahteva povezavo in je lastnik skupine

1. V ukazni vrstici `wpa_cli` podamo ukaz za izbiro metode WPS – tipka in ukaz za iskanje.

```
>set config_methods push_button
>p2p_find
```

2. Računalnik po iskanju vrne množico najdenih naprav in lastnosti.

```
>P2P-DEVICE-FOUND 02:08:22:96:c2:8d p2p_dev_addr=02:08:22:96:c2:8d
    pri_dev_type=10-0050F204-5 name='Prestigio4500' config_methods=0x188
    dev_capab=0x25 group_capab=0x0
```

3. V sistemskem grafičnem vmesniku za upravljanje povezav Wi-Fi Direct naprave Android je viden računalnik, z dotikom na ime računalnika pa se v `wpa_cli` ukazni vrstici računalnika prikaže povabilo za vzpostavljanje povezave s tipko ter se začne proces dogovarjanja o lastniku skupine.

```
>P2P-PROV-DISC-PBC-REQ 02:08:22:96:c2:8d p2p_dev_addr=02:08:22:96:c2:8d
    pri_dev_type=10-0050F204-5 name='Prestigio4500' config_methods=0x188
    dev_capab=0x27 group_capab=0x0
>P2P-GO-NEG-REQUEST 02:08:22:96:c2:8d dev_passwd_id=4
```

4. Odgovorimo z ukazom za povezovanje s parametri naslov MAC naprave Android in metoda WPS – tipka.

```
>p2p_connect 02:08:22:96:c2:8d pbc
```

5. Po dogovoru o lastniku skupine se ustvari skupina, katere lastnik je naprava Android, računalnik pa odjemalec P2P.

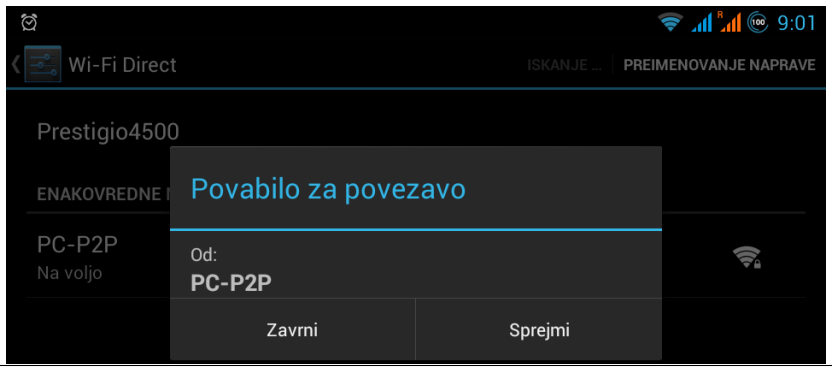
```
>P2P-GO-NEG-SUCCESS
>P2P-GROUP-FORMATION-SUCCESS
>P2P-GROUP-STARTED p2p-wlan0-0 client ssid="DIRECT-8u-Prestigio4500"
    freq=2437 psk=574a8c72f967b6da494f4ef0bf0b9212...
    go_dev_addr=02:08:22:96:c2:8d
```

Vzpostavljanje povezave, kjer ima računalnik večje število o lastništvu skupine, oziroma je lastnik, ne bomo posebej obravnavali, ker je postopek enak. Moramo omeniti, da je naprava Android prekinjala povezavo, ko na računalniku, ki je lastnik skupine, ni tekel strežnik DHCP, kar ni v skladu s standardom, zaradi tega pa takšnega obnašanja naprave Android ne bomo obravnavali kot napako v delovanju.

Ko računalnik zahteva povezovanje je postopek enak, edina razlika je da ni tretjega koraka. Po četrtem koraku se na napravi Android prikaže pogovorno okno povabila, na katerega odgovorimo potrdilno če se strinjamo z zahtevo računalnika za povezovanje. Pogovorno okno je v bistvu ekvivalentno tretjemu koraku, oziroma je grafično implementiran tretji korak (slika 4.5).

Ukaz za pridruževanje skupini je enak kot pri ustvarjanju skupine, oziroma povezovanju, na koncu ukaza pa se doda parameter `join`.

```
>p2p_connect 02:08:22:96:c2:8d pbc join
```




```
>P2P-PROV-DISC-PBC-REQ 02:08:22:96:c2:8d
    p2p_dev_addr=02:08:22:96:c2:8d pri_dev_type=10-0050F204-5
    name='Prestigio4500' config_methods=0x188 dev_capab=0x27
    group_capab=0x0
```

Slika 4.5: Android pogovorno okno za potrditev povabila Wi-Fi Direct povezave s tipko, ki je ekvivalentno tretjemu koraku v katerem računalnik dobi zahtevo, oziroma povabilo za povezovanje od naprave Android

4.3.2 Vzpostavljanje povezave z metodo WPS – PIN

Naprava Android zahteva povezavo, je lastnik skupine in prikazuje PIN

1. V ukazni vrstici `wpa_cli` podamo ukaz za izbiro metode WPS PIN – prikaži (angl. display), ki zahteva od naprave Android prikaz števila PIN in ukaz za iskanje.

```
>set config_methods display
>p2p_find
```

2. Računalnik po iskanju vrne množico najdenih naprav in lastnosti.

```
>P2P-DEVICE-FOUND 02:08:22:96:c2:8d p2p_dev_addr=02:08:22:96:c2:8d
    pri_dev_type=10-0050F204-5 name='Prestigio4500' config_methods=0x188
    dev_capab=0x25 group_capab=0x0
```

3. V sistemskem grafičnem vmesniku za upravljanje povezav Wi-Fi Direct naprave Android je viden računalnik, z dotikom na ime računalnika pa se prikaže pogovorno okno povabila s številom PIN (slika 4.6b). V `wpa_cli` ukazni vrstici računalnika se prikaže povabilo za vzpostavljanje povezave z vpisovanjem prikazanega števila PIN ter začne proces dogovarjanja o lastniku skupine.

```
>P2P-PROV-DISC-ENTER-PIN 02:08:22:96:c2:8d p2p_dev_addr=02:08:22:96:c2:8d
    pri_dev_type=10-0050F204-5 name='Prestigio4500' config_methods=0x188
    dev_capab=0x27 group_capab=0x0
>P2P-GO-NEG-REQUEST 02:08:22:96:c2:8d dev_passwd_id=5
```

4. Odgovorimo z ukazom za povezovanje s parametri naslov MAC naprave Android in prikazanega števila PIN.

```
>p2p_connect 02:08:22:96:c2:8d 81806669
```

5. Po dogovoru o lastniku skupine se ustvari skupina katere lastnik je naprava Android, računalnik pa odjemalec P2P.

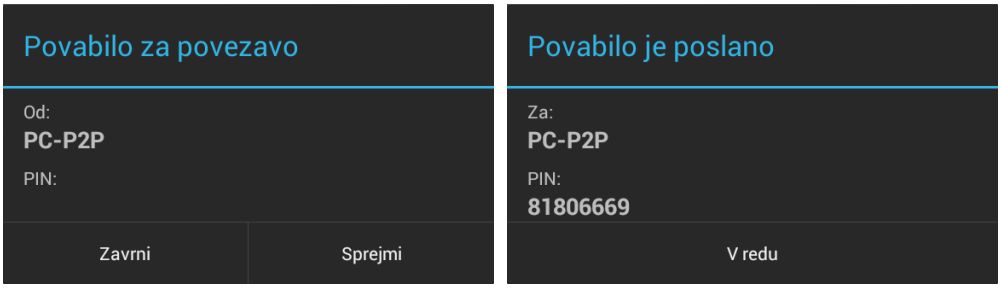
```
>P2P-GO-NEG-SUCCESS
>P2P-GROUP-FORMATION-SUCCESS
>P2P-GROUP-STARTED p2p-wlan0-0 client ssid="DIRECT-GX-Prestigio4500"
    freq=2437 psk=083e22d7ed78a86ddf2d9de66436e56eb8ce6c3...
    go_dev_addr=02:08:22:96:c2:8d
```

V primeru ko računalnik zahteva povezavo mora vnaprej vedeti PIN. Dodatni koraki za poizvedbo PIN so:

1. Zahteva računalnika za prikazovanjem števila PIN.

```
>p2p_prov_disc 02:08:22:96:c2:8d display
```

2. Naprava Android prikaže pogovorno okno za potrditev povezovanja s prikazovanjem števila PIN (slika 4.6a), po potrdilnem odgovoru pa se prikaže novo pogovorno okno povabila s številom PIN (slika 4.6b). Nadaljujemo s četrtim korakom.



(a) Dovoljenje napravi za povezovanje s prikazom števila PIN

(b) Naključno število PIN ki se vpiše v drugo napravo

Slika 4.6: Pogovorna okna pri povezovanju z metodo WPS – PIN prikaži

Naprava Android zahteva povezavo, je lastnik skupine in zahteva PIN

1. V ukazni vrstici `wpa_cli` podamo ukaz za izbiro metode WPS PIN – vpiši (angl. keypad), ki zahteva od naprave Android vpis števila PIN in ukaz za iskanje.

```
>set config_methods keypad
>p2p_find
```

2. Računalnik po iskanju vrne množico najdenih naprav in lastnosti.

```
>P2P-DEVICE-FOUND 02:08:22:96:c2:8d p2p_dev_addr=02:08:22:96:c2:8d
    pri_dev_type=10-0050F204-5 name='Prestigio4500' config_methods=0x188
    dev_capab=0x25 group_capab=0x0
```

3. V sistemskem grafičnem vmesniku za upravljanje povezav Wi-Fi Direct naprave Android je viden računalnik, z dotikom na ime računalnika pa se prikaže pogovorno okno povabila za vnos številke PIN. V `wpa_cli` ukazni vrstici računalnika se prikaže povabilo za vzpostavljanje povezave s prikazom števila PIN.

```
>P2P-PROV-DISC-SHOW-PIN 02:08:22:96:c2:8d 39063564
    p2p_dev_addr=02:08:22:96:c2:8d pri_dev_type=10-0050F204-5
    name='Prestigio4500' config_methods=0x188 dev_capab=0x27 group_capab=0x0
```

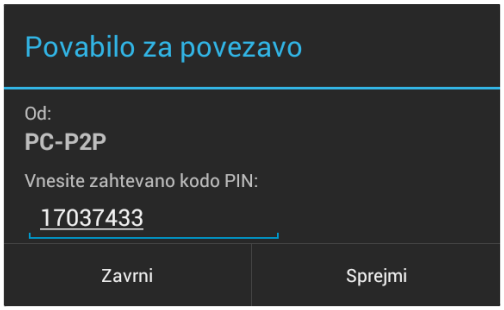
4. Odgovorimo z ukazom za povezovanje s parametri: naslov MAC naprave Android, metoda PIN in prikaz PIN. Ukazna vrstica `wpa_cli` prikaže naključno število PIN.

```
>p2p_connect 02:08:22:96:c2:8d pin display
>17037433
```

5. Prikazani PIN iz prejšnjega koraka vpišemo v napravo Android (slika 4.7).

6. Po dogovoru o lastniku skupine se ustvari skupina katere lastnik je naprava Android, računalnik pa odjemalec P2P.

```
>P2P-GO-NEG-SUCCESS
>P2P-GROUP-FORMATION-SUCCESS
>P2P-GROUP-STARTED p2p-wlan0-6 client ssid="DIRECT-KC-Prestigio4500"
    freq=2437 psk=aa329053b4791d160e7f42a026393d0ba31cc7...
    go_dev_addr=02:08:22:96:c2:8d
```



Slika 4.7: Pogovorno okno povabila s poljem za vnos števila PIN

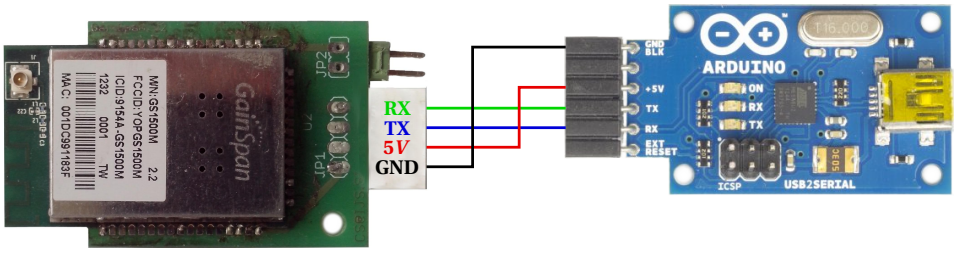
V primeru ko računalnik zahteva povezavo se vpiše ukaz iz četrtega koraka, naprava Android pa prikaže pogovorno okno povabila za vnos števila PIN. Število PIN se lahko tudi poljubno definira tako, da se namesto parametra pin vpiše pravilno število PIN.

```
>p2p_connect 02:08:22:96:c2:8d 12345670 display
```

.

4.4 Povezava Wi-Fi Direct z modulom GainSpan

Testno vzpostavljanje povezav Wi-Fi Direct z modulom smo krmilili s pomočjo vmesniškega vezja Arduino USB2Serial, ki omogoča komunikacijo UART preko vmesnika USB. Povezavo modulov prikazuje slika 4.8, modul USB2Serial pa smo povezali z računalnikom. Modulu GainSpan pošiljamo ukaze iz aplikacije CuteCom, kateri moramo najprej nastaviti enake lastnosti komunikacije UART kot so pri modulu GainSpan. Privzeta nastavitev hitrosti vmesnika UART je 9600bps, naenkrat se prenaša 8 bitov, en stop bit pa označuje konec sporočila.



Slika 4.8: Povezava modulov GainSpan 1500M in USB2Serial

Modul GainSpan podpira množico ukazov ki omogočajo nastavitve različnih parametrov delovanja modula. Ker je ukazov veliko, se bomo omejili na tiste ki so potrebni pri vzpostavljanju povezave Wi-Fi Direct [8]. Ukaze in zaporedje ukazov pri vzpostavljanju povezav bomo uporabili pri programiranju mikrokrmilne knjižnice, zaradi tega pa moramo najti najboljšo kombinacijo le-teh ter vlogo modula v omrežju.

4.4.1 Ukazi za vzpostavljanje povezave Wi-Fi Direct

Ukazi modula so skladni s standardom ITU V.25ter, kjer se večina ukazov začne s predpono `AT`. Modul se na ukaze standardno odziva z odgovorom `OK` v primeru, ko se ukaz pravilno izvede ali z napako `ERROR` po kateri sledi opis napake. V odvisnosti od ukaza se izpisujejo dodatna sporočila, ki omogočajo pravilen odziv, oziroma izbiro naslednjega ukaza.

AT+WRXACTIVE=1

Ukaz onemogoča modulu varčevanje z energijo, ker želimo da bo modul vedno viden in imel najkrajše možne čase odzivnosti. Ukaz vrača standardne odgovore.

AT+P2PSETDEV

Ukaz `AT+P2PSETDEV=<1>,<2>,<3>,<4>,<5>,<6>` omogoča nastavitev lastnosti naprave P2P. Sprejema 6 parametrov:

1. število o lastniku skupine ki je med 0 in 15,
2. izbira dovoljenih kanalov:
   - 81 – 11g kanali od 1 do 13,
   - 82 – 11g kanali od 1 do 14,
   - 115 – 11a kanali od 36 do 48,
   - 124 – 11a kanali od 149 do 161,
3. kanal na katerem modul posluša zahteve naprav P2P, sprejema števila od 1 do 14,
4. kanal na katerem modul deluje, sprejema števila od 1 do 14,
5. izbira metod WPS, izračunajo se iz kode 3.5.
   - 0008 – prikaži,
   - 0100 – vpiši,
   - 0080 – tipka,
6. država uporabe modula, veljavni parametri so US, JP, FR, GB, AU, GE, IN.

Ukaz vrača standardne odgovore.

AT+P2PSETWPS

Ukaz `AT+P2PSETWPS=<1>,<2>,<3>,<4>,[]...[]` omogoča nastavite lastnosti WPS. Potrebuje minimalno 4 parametre:

1. ime naprave vidno pri iskanju dolžine do 32 črk,
2. primarni tip naprave definiran v protokolu WPS dolžine 2 bajta,
3. sekundarni tip naprave definiran v protokolu WPS dolžine 2 bajta,
4. uuid število dolžine 16 bajtov ki se uporablja pri protokolu WPS za enolično določanje WPS transakcij,
- dodatni primarni in sekundarni tipi naprav, podpira do 5 parov, pred prvim parom se mora nahajati število dodatnih parov.

Ukaz vrača standardne odgovore.

AT+WM

Ukaz `AT+WM=<način>` izbira način delovanja, oziroma vlogo modula v omrežju:

0 odjemalec,

1 ad hoc,

2 brezžična dostopna točka,

3 Wi-Fi Direct.

Ko se zahteva način Wi-Fi Direct, ukaz vrne napako če modul nima nastavljenih nastavitev P2P, oziroma se mora izvesti po ukazih `AT+P2PSETWPS` in `AT+P2PSETDEV`. Ukaz vrača standardne odgovore.

AT+P2PLISTEN

Z ukazom `AT+P2PLISTEN=<čas>` se modul nastavi v stanje poslušanja po zahtevah naprav za povezovanje. Čas je v sekundah, če ni določen je naprava vedno v stanju poslušanja.

V stanju poslušanja, po sprejemu zahteve za povezavo, modul preko vmesnika UART vrača naslednja sporočila:

1. zahteva povezovanja s pomočjo tipke:

```
p2p-prov-disc-req pbc <naslov MAC pošiljatelja>,<izvor>,<primarni tip
    WPS>,<primarni podtip WPS>,<sekundarni tip WPS>,<sekundarni podttip
    WPS>,<ime P2P>,<kanal>,<0080>,<lastnosti naprave>,<lastnosti skupine>
```

2. zahteva za prikaz števila PIN:

```
p2p-prov-disc-req display-pin <PIN> <naslov MAC pošiljatelja>,<izvor>,
    <primarni tip WPS>,<primarni podtip WPS>,<sekundarni tip
    WPS>,<sekundarni podttip WPS>,<ime P2P>,<kanal>,<0008>,<lastnosti
    naprave>,<lastnosti skupine>
```

3. zahteva za vpis števila PIN:

```
p2p-prov-disc-req enter-pin <naslov MAC pošiljatelja>, <izvor>,<primarni
    tip WPS>,<primarni podtip WPS>, <sekundarni tip WPS>,<sekundarni
    podttip WPS>,<ime P2P>,<kanal>,<0100>,<lastnosti naprave>,<lastnosti
    skupine>
```

Ukaz vrača standardne odgovore.

AT+P2PFIND

Z ukazom `AT+P2PFIND=<čas>,<tip>` se modul nastavi v način iskanja, oziroma pošilja iskalne okvirje. V tem načinu lahko odgovarja na zahteve po povezavah, ki so enaki kot pri ukazu `AT+P2PLISTEN`, čeprav so možnosti majhne zaradi spreminjanja kanalov. Parameter `<čas>` določa čas iskanja v sekundah, če ni določen je naprava vedno v stanju iskanja. Parameter `<tip>` definira dva načina iskanja:

0 – iskanje po družabnih kanalih 1, 6, 11,

1 – iskanje po vseh kanalih.

V času iskanja modul vrača informacije o najdenih naprav P2P:

```
p2p-dev-found <naslov MAC pošiljatelja>,<izvor>,<primarni tip WPS>,<primarni
    podtip WPS>,<sekundarni tip WPS>,<sekundarni podttip WPS>,<ime
    P2P>,<kanal>,<WPS metoda>,<lastnosti naprave>,<lastnosti skupine>
```

Iskanje se ustavi z ukazom `AT+P2PSTOPFIND`, iskanje pa ni mogoče, ko je modul lastnik skupine. Ukaz vrača standardne odgovore.

AT+P2PGRPFORM

Ukaz se uporablja za povezovanje, oziroma ustvarjanje skupine z dogovorom o lastniku skupine. Sprejema 7 parametrov:

1. naslov MAC naprave s katero se povezuje,
2. kanal skupine,
3. način WPS,

    4 – povezovanje s tipko,

    2 – prikaz števila PIN na modulu,

3 – vnos števila PIN v modulu,

4. število PIN,
5. število o lastniku skupine,
6. če je 1, modul je v stanju poslušanja in lahko odgovarja na zahteve po ustvarjanju skupine,
7. če je 1, skupina je obstojna.

V odvisnosti od vloge modula v skupini, po uspešni vzpostavitvi skupine vrača dve sporočili:

1. modul je lastnik skupine

```
p2p-go-neg-complete GO,<ssid>,<kanal>,<naslov MAC lastnika skupine>,<ključ>
```

2. modul je odjemalec

```
p2p-go-neg-complete client,<ssid>,<kanal>,<naslov MAC lastnika
    skupine>,<ključ>
```

V primeru napake pri vzpostavljanju skupine vrne `p2p-go-neg <razlog>`. Ukaz vrača standarde odgovore.

AT+P2PGOSTART

Ukaz `AT+P2PGOSTART=<1>,[2],[3],[4]` omogoča ustvarjanje avtonomne skupine. Parametri ukaza so:

1. kanal delovanja skupine,
2. dodatek k imenu skupine,
3. če je 1, je skupina obstojna, v primeru GainSpan modula se skupina ohranja samo med delovanjem modula,
4. identifikacijska številka skupine.

Obvestilo o uspešnem ustvarjanju avtonomne skupine je:

```
p2p-go-started DIRECT-<dinamično ime +dodatek iz drugega
    parametra>,<kanal>,<naslov MAC lastnika skupine>, <ključ>,<PIN>
```

Moramo omeniti, da se ključ skupine ne more definirati, ampak je generiran iz imena iz ukaza `AT+P2PSETWPS`. Dokler je ime enako, je ključ vedno enak, PIN pa se vsakič spremeni. Čeprav v dokumentaciji ni omenjeno, se število PIN po ustvarjanju skupine lahko spremeni z ukazom `AT+WWPS=2,<PIN>`, ki je opisan samo pri delovanju modula v načinu dostopne točke. Ukaz vrača standarde odgovore.

4.4.2 Vzpostavljanje povezav Wi-Fi Direct z računalnikom

Računalnik zahteva povezavo, ni lastnik skupine, metoda WPS tipka

1. Najprej smo iz aplikacije za serijsko komunikacijo CuteCom podali vse potrebne ukaze za nastavitev modula v Wi-Fi Direct način in ga postavili v stanje poslušanja.

```
AT+WRXACTIVE=1
AT+P2PSETDEV=15,81,6,6,0080,FR
AT+P2PSETWPS=GAINSPAN-P2P,000a,0005,11223344556677881122334455667733
AT+WM=3
AT+P2PLISTEN=,
```

2. Modul je viden računalniku po iskanju.

```
P2P-DEVICE-FOUND 02:1d:c9:91:18:3f p2p_dev_addr=02:1d:c9:91:18:3f
    pri_dev_type=10-0050F204-5 name='GAINSPAN-P2P' config_methods=0x80
    dev_capab=0x21 group_capab=0x0
```

3. V ukazni vrstici `wpa_cli` vpišemo ukaz za povezovanje z metodo tipka in čakamo da modul sprejme zahtevo za povezavo z metodo WPS tipka. Po dolgem čakanju se ni nič izpisalo, na računalniku pa se je izpisalo sporočilo o napaki pri dogovarjanju o lastniku skupine.

```
>p2p_connect 02:1d:c9:91:18:3f pbc
>P2P-GO-NEG-FAILURE status=-1
```

4. Naslednje kar smo poskusili je bilo vpisovanje ukazov za povezovanje s tipko na obeh straneh. Povezava je bila uspešno vzpostavljena.

```
AT+P2PGRPFORM=00:22:43:51:67:17,,4,,15,1,0
p2p-go-neg-complete GO,DIRECT-38,6,02:1d:c9:91:18:3f,BFlwV0ZM
```

Da bi lahko programsko vedeli, kdaj moramo vzpostaviti skupino, moramo dobiti obvestilo iz ukaza `AT+P2PLISTEN`. Zaradi tega smo poskusili z ukazom `p2p_prov_disc` z metodo tipka, modul pa je uspešno sprejel obvestilo računalnika za povezovanjem:

1. Ukaz računalnika za povabilo povezovanja s tipko modulu GainSpan.

```
>p2p_prov_disc 02:1d:c9:91:18:3f pbc
>P2P-PROV-DISC-PBC-RESP 02:1d:c9:91:18:3f
```

2. Modul je sprejel vabilo za povezovanje s tipko.

```
p2p-prov-disc-req pbc
    00:22:43:51:67:17,00:22:43:51:67:17,000a,0050,f204,0005,
    PC-P2P,0080,25,00
```

Ko smo ukaz `p2p_prov_disc` preizkusili z metodami WPS PIN, ko je modul nastavljen, da podpira samo tipko, je bil odgovor na zahtevo negativen (`P2P-PROV-DISC-FAILURE`). V načinu, ko je modul lastnik skupine, se je avtomatično zagnal strežnik DHCP, v kar smo se

prepričali tako, da smo na računalniku dobili naslov IP s pomočjo `dhcpcd` odjemalca DHCP na vmesniku P2P.

Računalnik zahteva povezavo, ni lastnik skupine, zahteva prikaz PIN

1. Inicializiramo nastavitve modula za prikaz števila PIN in poslušamo za zahtevami.

```
AT+WRXACTIVE=1
AT+P2PSETDEV=15,81,6,6,0008,FR
AT+P2PSETWPS=GAINSPAN-P2P,000a,0005,11223344556677881122334455667733
AT+WM=3
AT+P2PLISTEN=,
```

2. Ko je modul viden računalniku, na računalniku izvedemo ukaz za prikaz števila PIN.

```
>p2p_prov_disc 02:1d:c9:91:18:3f display
>P2P-PROV-DISC-ENTER-PIN 02:1d:c9:91:18:3f
```

3. Modul sprejme zahtevo za povezovanje. Izvedemo ukaz za povezovanje s prikazovanjem izbranega števila PIN.

```
p2p-prov-disc-req display-pin 00000000
    00:22:43:51:67:17,00:22:43:51:67:17,000a,0050,f204,0005,
    PC-P2P,0008,25,00
AT+P2PGRPFORM=00:22:43:51:67:17,,2,12345670,15,1,0
p2p-go-neg-complete GO,DIRECT-rB,6,02:1d:c9:91:18:3f,7JOUmysi
```

4. Na računalniku izvedemo ukaz za povezovanje s prej izbranim, oziroma prikazanim številom PIN. Naprave so se uspešno povezale.

```
>p2p_connect 02:1d:c9:91:18:3f 12345670
>P2P-FIND-STOPPED
>P2P-GO-NEG-SUCCESS
>P2P-GROUP-STARTED p2p-wlan0-31 client ssid="DIRECT-rB" freq=2437
    passphrase="7JOUmysi" go_dev_addr=02:1d:c9:91:18:3f
```

Računalnik zahteva povezavo, ni lastnik skupine, prikazuje PIN

1. Inicializiramo nastavitve modula za vnos števila PIN in poslušamo zahteve.

```
AT+WRXACTIVE=1
AT+P2PSETDEV=15,81,6,6,0100,FR
AT+P2PSETWPS=GAINSPAN-P2P,000a,0005,11223344556677881122334455667733
AT+WM=3
AT+P2PLISTEN=,
```

2. Ko je modul viden računalniku, na računalniku izvedemo ukaz, ki od modula zahteva vpis števila PIN.

```
>p2p_prov_disc 02:1d:c9:91:18:3f keypad
>P2P-PROV-DISC-SHOW-PIN 02:1d:c9:91:18:3f 74139682
```

3. Modul sprejme zahtevo za povezovanje. Izvedemo ukaz za povezovanje z vpisom števila PIN.

```
p2p-prov-disc-req enter-pin 00:22:43:51:67:17,00:22:43:51:67:17,000a,0050,
    f204,0005,PC-P2P,0100,25,00
AT+P2PGRPFORM=00:22:43:51:67:17,,3,12345670,15,1,0
p2p-go-neg-complete GO,DIRECT-NT,6,02:1d:c9:91:18:3f,WayMaBvQ
```

4. Na računalniku izvedemo ukaz za povezovanje s prikazom števila PIN. Naprave so se uspešno povezale.

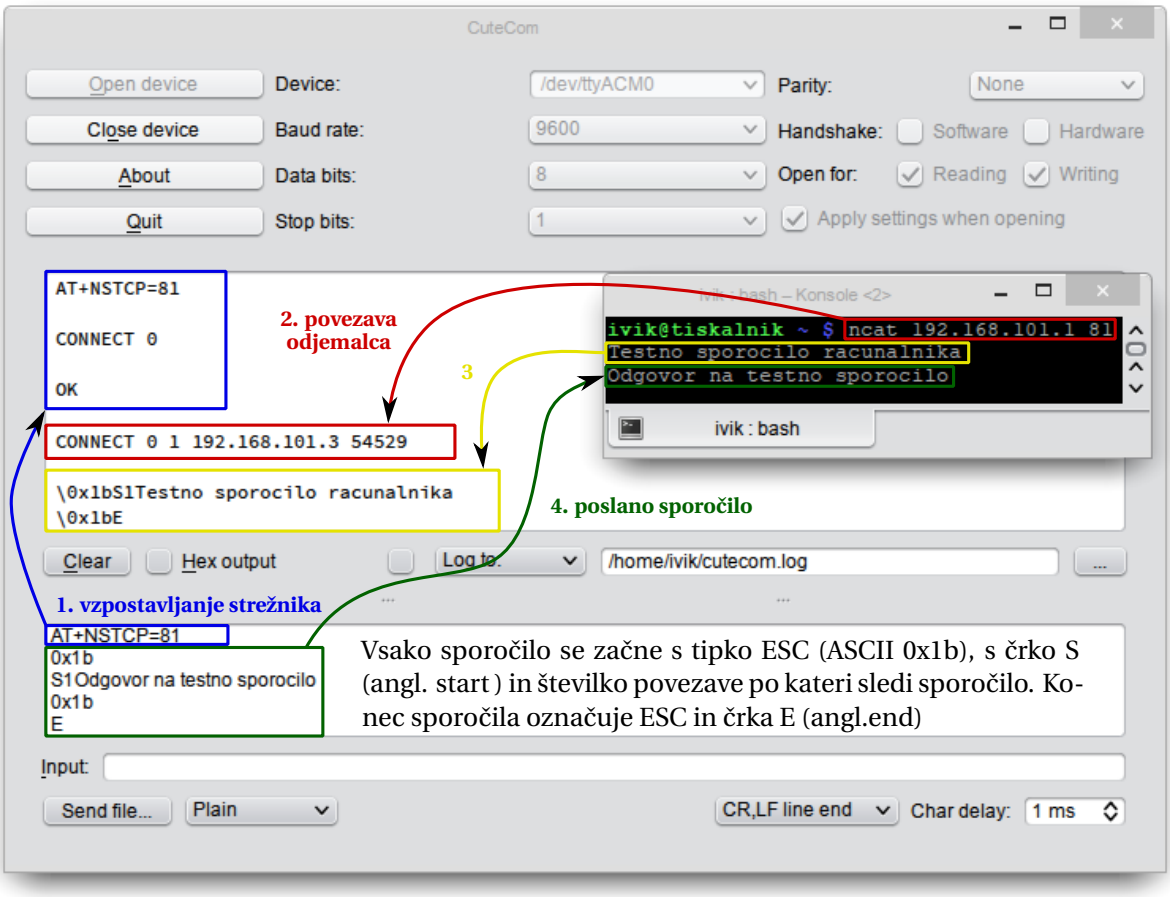
```
>p2p_connect 02:1d:c9:91:18:3f 12345670 display
>P2P-GO-NEG-SUCCESS
>P2P-GROUP-STARTED p2p-wlan0-32 client ssid="DIRECT-NT" freq=2437
    passphrase="WayMaBvQ" go_dev_addr=02:1d:c9:91:18:3f
```

4.4.3 Vzpostavitev strežnika TCP/IP in testiranje povezave

Sama povezava računalnika in modula GainSpan 1500M na fizični plasti nima smisla če se ne vzpostavi komunikacija na višji plasti. Komunikacijo med modulom in računalnikom smo vzpostavili po metodologiji odjemalec–strežnik (angl. client–server) s protokolom prenosne plasti TCP/IP ISO/OSI referenčnega modela. Sporočila preko protokola TCP se zaradi vzpostavljene povezave med odjemalcem in strežnikom prenašajo zanesljivo v obe smeri, so brez napak, brez podvojevanja in so v pravilnem vrstnem redu. Odjemalec je vsak računalnik, na daljavo povezan s strežniškim računalnikom, ki mu zagotavlja določeno storitev. Praviloma se na en strežnik lahko poveže več odjemalcev. Pri vzpostavitvi povezave, mora odjemalec vedeti naslov IP in vrata (angl. port) strežnika, strežnik pa mora "poslušati" na tem naslovu in vratih.

Strežnik TCP/IP modula se vzpostavlja z ukazom `AT+NSTCP=<vrata>`. Pred vzpostavitvijo mora biti modul povezan v enem od načinov iz ukaza `AT+WM`. Obvestilo o uspešni vzpostavitvi strežnika je `CONNECT <SID>`, kjer `SID` določa identifikacijsko število strežnika, po prvi vzpostavitvi je 0. Povezava odjemalca s strežnikom se sporoči s `CONNECT <SID> <CID>`, kjer je `<CID>` od 1 do F in določa identifikacijsko številko odjemalca, oziroma povezave. Podobno obvestilo je pri prekinitvi povezave z odjemalcem `DISCONNECT <SID> <CID>`. Da bi se sporočila TCP/IP razlikovala od obvestil modula, imajo obliko `<ESC>S<CID><sporočilo><ESC>E`, kjer jer `<ESC>` ASCII koda tipke Escape, `0x1b` v šesnajstiški obliki. Enak format se uporablja tudi pri pošiljanju. Pri neuspešnem pošiljanju sporočila TCP/IP, ko se povezava prekine pred obvestilom o prekinitvi, se sporoči z `ERROR: SOCKET FAILURE <CID>`.

Strežnik smo vzpostavili na vratih 81. Vlogo odjemalca računalnika smo vzpostavili s pomočjo ukaznega programa `ncat` ki je del programskega orodja nmap [10]. Postopek vzpostavljanja povezave prikazuje slika 4.9.



Slika 4.9: Vzpostavljanje strežnika na modulu GainSpan 1500M, povezovanje s strežnikom in izmenjava sporočil TCP/IP

4.4.4 Povezovanje z napravo Android

Postopek povezovanja naprave Android in modula poteka podobno kot pri povezovanju z računalnikom. Vsi ukazi modulu so identični, pojavila se je pa težava pri pridruževanju skupini. Ko smo prekinili povezavo med modulom in napravo Android, modul še vedno vzdržuje skupino, naprava Android pa se ni hotela pridružiti skupini. Nobenega sporočila UART modula nismo sprejeli pri ponovnem povezovanju, kot tudi ne pogovornega okna s številom PIN. Domnevamo da je napaka pri napravi Android, ki storitvi `wpa_supplicant` ne pošlje ukaza za povezovanje s parametrom `join`, oziroma se ne zaveda, da se pridružuje skupini. Računalnik se je brez problemov povezoval z že ustvarjeno skupino.

4.5 Mikrokrmilna knjižnica Arduino

Po vseh testiranjih povezovanja se je za najboljšo vlogo modula GainSpan 1500M v omrežju izkazalo:

1. da je lastnik skupine,
2. da prikazuje PIN,
3. da je strežnik.

Razlogi za to so naslednji:

- ko je modul lastnik skupine, vsak član skupine ve kdo je lastnik skupine, kar tudi pomeni, da ve na katerem naslovu IP je strežnik,
- ko modul ni lastnik skupine, so vse ostale naprave, ki se pridružijo skupini odvisne od lastnika skupine in lahko komunicirajo s strežnikom le posredno preko lastnika skupine toliko časa, kolikor je lastnik skupine pripravljen vzdrževati skupino,
- ko modul ni lastnik skupine, morajo vse ostale naprave poiskati naslov strežnika, ker pa modul ne podpira oglaševanja storitve (angl. service discovery), ga druge naprave v omrežju ne morejo na enostaven način poiskati,
- operacijski sistem Android ne podpira določanja številke PIN pri prikazovanju, vpis številke v modul pa bi zahtevalo tipkovnico in dodatno programsko kodo mikrokrmilnika za branje števila PIN, z vpisovanjem pa bi se tudi izgubilo na sami mobilnosti ker bi bil potreben fizični dostop do modula,
- povezovanje s pritiskom tipke bi tudi zmanjšalo mobilnost, če pa bi bila potrditev programsko implementirana, ne moremo enostavno določiti kdo ima pravico do povezave,
- avtonomna skupina ni prišla v poštev, ker ni možno določiti na katerem kanalu delujejo postaje, ki se bodo povezovale, z dogovorom o skupini pa se kanal določi s pomočjo postaje s katero se vzpostavlja skupina, s tem pa se postaji omogoči neovirana komunikacija z dostopno točko s katero je povezana.

Funkcionalnost Arduino mikrokrmilne knjižnice lahko razdelimo na štiri dele:

1. pošiljanje ukazov in branje odziva,
2. inicializacija nastavitev,
3. upravljanje vzpostavljanja povezave Wi-Fi Direct,
4. branje in pošiljanje sporočil TCP/IP.

4.5.1 Pošiljanje ukazov in branje odziva

Ukazi UART se v Arduino razvojnem okolju pošiljajo s pomočjo razreda `Serial` s funkcijami:

- `Serial.write()` – omogoča pošiljanje podatkov v obliki byte,
- `Serial.println()` ali `Serial.print()` – omogoča pošiljanje črk ali besed z ali brez znaka za konec vrstice.

Tako se npr. ukaz modula `AT`, ki preverja če je vmesnik UART pravilno inicializiran na obeh straneh, pošlje iz Arduino razvojnega okolja z ukazom `Serial.println("AT")`. Ukazi modula se morajo končati z novo vrstico, ki jo avtomatično doda ukaz `Serial.println()`, ali če se na koncu besede doda črka `'\r'` ali `'\n'`.

Sporočila UART se berejo z ukazom `Serial.read()`, ki prebere en znak, preverjanje obstoja podatkov pa omogoča funkcija `Serial.available()`. Da bi prebrali celotno vrstico, moramo v zanki brati in shranjevati vse črke do znaka za konec vrstice (koda 4.2).

Programska koda 4.2: Branje vrstice

```
String P2PClass::readline(void) {
  String buf;
  char inChar;
  bool end = false;

  while (!end) {
    if (Serial.available()) {
      inChar = Serial.read();
      if ((inChar == '\r') || (inChar == '\n')) {
        if ((buf.length() > 0) && (inChar == '\n'))
          end = true;
      }
      else buf += inChar;
    }
  }
  return buf;
}
```

Pred vsakim ukazom moramo vse podatke vmesnika UART prebrati zaradi zagotovitve, da je naslednji odziv res odziv zadnjega ukaza (koda 4.3).

Programska koda 4.3: Branje vseh podatkov vmesnika UART pred naslednjim ukazom

```
void P2PClass::flush() {
  while (Serial.available())
    Serial.read();
}
```

Odziv modula se primerja s standardnimi odgovori (koda 4.4). V primeru, ko se ukaz uspešno izvede, se lahko nadaljuje z naslednjim ukazom.

Programska koda 4.4: Primerjanje odziva s standardnim odgovorom

```
uint8_t P2PClass::checkResponse() {
  String buf;
  while (1) {
    buf=readline();
    if (buf == "OK")
      return 1;
    else if (buf.startsWith("ERROR"))
      return 0;
  }
}
```

4.5.2 Inicializacija nastavitev

Knjižnici, ki je vključena v programsko okolje Arduino (koda 4.5, vrstica 1), se morajo v Arduino funkciji `setup()` podati parametri delovanja modula in mikrokrmilne platforme Arduino.

Programska koda 4.5: Inicializacija knjižnice v programskom okolju Arduino

```
#include <P2P.h>                      // vključitev knjižnice
                                      // v razvojno okolje Arduino
void setup() {
  P2P.name="GAINSPAN-P2P";            // ime modula vidno pri iskanju
  P2P.PIN="12345670";                 // PIN
  P2P.server_addr="192.168.105.1";    // naslov IP
  P2P.server_port=81;                 // vrata strežnika
  P2P.uart_speed=0;                   // 0=9600bps, 1=115200bps
  pinMode(8, OUTPUT);                 // na nožico 8 je priklopljena LED dioda
  if (!P2P.init()) return;

  digitalWrite(8, HIGH);              // vklop LED diode po uspešni inicializaciji
  while (!P2P.connect());             // vzpostavljanje povezave Wi-Fi Direct
}
```

Funkcija `init()` (koda 4.6) inicializira komunikacijo UART mikrokrmilnika s pomočjo ukaza `Serial.begin(<hitrost>,<lastnost>)`:

- `<hitrost>` – število od 300 do 115200 (bps),
- `<lastnost>` – določa število naenkrat prenešenih bitov, paritetnost in število stop bitov, privzeta vrednost je 8N1.

Odziv modula se preverja z ukazom `Serial.println("AT")`. Preverjanje odziva je potrebno zaradi zagotovitve pravilne fizične povezave med napravami (slika 4.10) ter preverjanja pripravljenosti naprav za sprejemanje in pošiljanje sporočil UART. Čeprav bi lahko funkcijo `init()` vključili v funkcijo za povezovanje `connect()`, nam takšna razdelitev omogoča, da po uspešni inicializaciji lahko dodamo poljubno programsko kodo, npr. kodo za prižiganje LED diode, ki nam vizualno sporoči, če je komunikacija med moduli uspešno vzpostavljena.

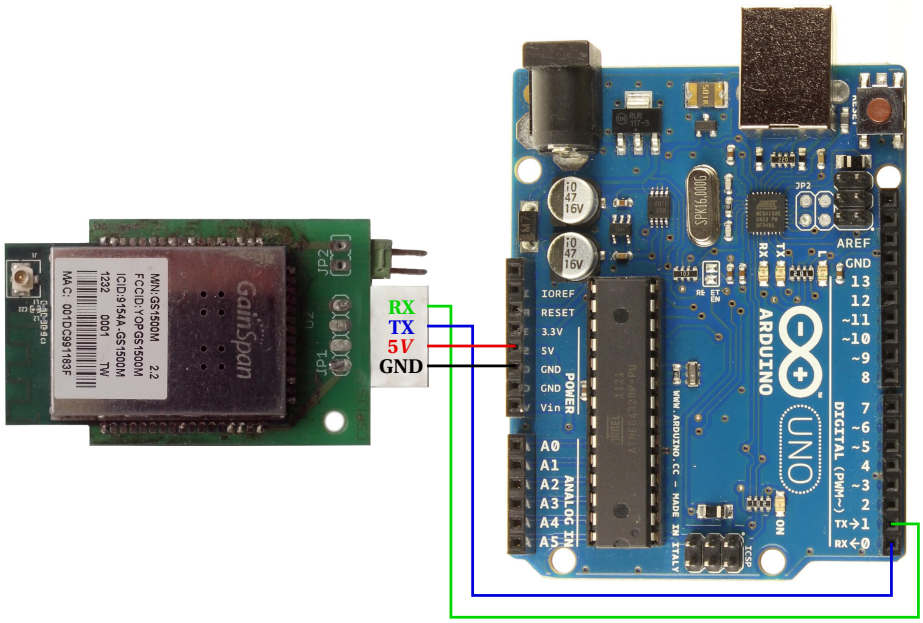
Programska koda 4.6: Funkcija `init()`

```
uint8_t P2PClass::init() {

  Serial.begin(9600);
  dataOnSock = 255;
  lastSocket = 255;

  flush();
  Serial.println(F("AT"));
  if (!checkResponse())
    return 0;


  if (uart_speed) {
    Serial.println(F("ATB=115200,8,n,1"));
    Serial.end();
    delay(1000);
    Serial.begin(115200);
    delay(1000);
    flush();
    Serial.println(F("AT"));
    if (!checkResponse())
      return 0;
  }

  return 1;
}
```

4.5.3 Upravljanje vzpostavljanja povezave Wi-Fi Direct

Ukazi za inicializacijo ter vzpostavljanje povezave Wi-Fi Direct (koda 4.13) so enaki kot pri vzpostavljanju povezave z računalnikom kadar modul prikazuje število PIN (poglavje 4.4.2). Ukazom se dodajo uporabniško definirani parametri, ki jih knjižnica predpisuje pri inicializaciji, ostali pa se dinamično določijo z branjem sporočil modula.

Ko je modul v stanju poslušanja, je knjižnica v stanju čakanja na obvestilo modula ki označuje zahtevo za vzpostavljanje skupine s prikazom števila PIN (koda 4.7, vrstica 1). Iz tega sporočila je potrebno prebrati naslov MAC naprave, ki zahteva povezavo, tako da se preberejo znaki sporočila med 40 in 57. Naslednji ukaz modulu je ukaz za ustvarjanje skupine



Slika 4.10: Povezava modula GainSpan 1500M in mikrokrmilne platforme Arduino UNO

(koda 4.7, vrstica 2), kateremu je prvi parameter naslov MAC iz prejšnjega sporočila. Po uspešno vzpostavljeni skupini se zažene strežnik TCP/IP.

Programska koda 4.7: Sporočilo za prikaz števila PIN iz katerega se prebere naslov MAC ki se uporabi pri ukazu za ustvarjanje skupine (vrstica 2)

```
1 p2p-prov-disc-req display-pin 00000000
      00:22:43:51:67:17,00:22:43:51:67:17,000a,0050,f204,0005,PC-P2P,0008,25,00
2 AT+P2PGRPFORM=00:22:43:51:67:17,,2,12345670,15,1,0
```

4.5.4 Branje in pošiljanje sporočil TCP/IP

Preverjanje obstoja sporočil

Uporabniški program s pomočjo funkcije knjižnice `available()` preveri obstoj sporočila TCP/IP (koda 4.8), v primeru potrdilnega odgovora pa je dovoljeno branje sporočila s funkcijo za branje.

Programska koda 4.8: Funkcija preverjanja obstoja sporočil TCP/IP

```
1 uint8_t P2PClass::available()
2 {
3   check();
4   return (dataOnSock != 255) ;
5 }
```

Iskanje začetka sporočila

Knjižnica preverja prihajajoča sporočila TCP/IP (koda 4.9) na način, da bere zaporedje znakov ki označujejo začetek sporočila TCP/IP, se ustavi ko je zaporedje pravilno in shrani številko povezave (vrstica 21).

Programska koda 4.9: Iskanje začetka sporočila TCP/IP

```
1  void P2PClass::check() {
2    if (!Serial.available())
3      return;
4
5    char temp;
6
7    while (dataOnSock == 255) {
8      if (Serial.available()) {
9        temp = Serial.read();
10       if (temp == 0x1b) {                        // 1. ESC
11         while(1) {
12           if (Serial.available()) {
13             temp = Serial.read();
14             if (temp == 0x53) {                  // 2. S
15               while(1) {
16                 if (Serial.available()) {
17                   temp = Serial.read();          // 3. številka povezave
18                   break;
19                 }
20               }
21               dataOnSock = lastSocket = temp;
22               break;
23             }
24             else break;
25           }
26         }
27       }
28     }
29     else break;
30   }
31 }
```

Branje sporočil TCP/IP

Uporabniška funkcija za branje sporočil TCP/IP (koda 4.10) sprejema dva parametra. Prvi je naslov tabele kamor se podatki zapisujejo, drugi pa določa število znakov, ki se morajo prebrati. Če je število večje od dolžine sporočila, se sporočilo prebere do konca. Ko je celotno sporočilo prebrano, funkcija nastavi spremenljivko (vrstica 21) na vrednost, ki pomeni da ni sporočila.

Programska koda 4.10: Funkcija branja sporočil TCP/IP

```
1  uint16_t P2PClass::read(char* char_buf, uint16_t len) {
2    uint16_t readLen = 0;
3
4    if (available()) {
5      if (!Serial.available())
6        return 0;
7
8      char temp1, temp2;
9
10     while(readLen < len) {
11       if (Serial.available()) {
12         temp1 = Serial.read();
13         if (temp1 == 0x1b) {                    // 1. ESC
14           while(1) {
15             if (Serial.available()) {
16               temp2 = Serial.read();
17               break;
18             }
19           }
20           if (temp2 == 0x45) {                  // 2. E
21             dataOnSock = 255;
22             break;
23           }
24           else {
25             if (readLen < (len-2)) {
26               char_buf[readLen++] = temp1;
27               char_buf[readLen++] = temp2;
28             }
29             else char_buf[readLen++] = temp1;
30           }
31         }
32         else {
33           char_buf[readLen] = temp1;
34           readLen++;
35         }
36       }
37     }
38   }
39   return readLen;
40 }
```

Pošiljanje sporočil TCP/IP

Funkcija za pošiljanje sporočil TCP/IP (koda 4.11) doda na začetku uporabniškega sporočila znake, ki modulu označujejo začetek sporočila TCP/IP in številko zadnje vzpostavljene povezave (vrstica 6). Zadnjo povezavo shranjujemo v dodatni spremenljivki ker se po branju celotnega sporočila številka povezave nastavi na 255, pri pošiljanju sporočila pa bi izgubili številko povezave odjemalca. Na konec sporočila doda zaporedje znakov, ki modulu označujejo konec sporočila TCP/IP.

Programska koda 4.11: Funkcija pošiljanja sporočil TCP/IP

```
1  uint16_t P2PClass::send(const char* message) {
2    if ((message[0] == '\r') || (message[0] == 0)){}
3    else {
4      Serial.write(0x1b);            // 1. ESC
5      Serial.write(0x53);            // 2. S
6      Serial.write(lastSocket);      // 3. številka odjemalca
7      Serial.write(message);         // 4. sporočilo
8      Serial.write(0x1b);            // 5. ESC
9      Serial.write(0x45);            // 6. E
10   }
11   delay(10);
12   return 1;
13 }
```

Uporaba funkcij

Opisane funkcije uporabljamo v glavni zanki `loop()` Arduino programa na način kot ga prikazuje koda 4.12. Primer prikazuje ponavljajoči strežnik (angl. echo server), ki sprejeta sporočila TCP/IP pošilja nazaj odjemalcu. Če je v sporočilu številka 1, se prižge analogni izhod mikrokrmilnika na katerega je povezana LED dioda, če pa je 0, pa se izklopi.

Programska koda 4.12: Uporaba funkcij za branje in pošiljanje sporočil TCP/IP v razvojem okolju Arduino

```
1  void loop() {
2    char temp[9];
3    memset(temp,0,sizeof(temp));
4    if (P2P.available()) {
5      P2P.read(temp,9);      // branje sporočila v spremenljivko temp
6      P2P.send(temp);        // pošiljanje sporočila
7      for (int i=0; i<9; i++) {
8        if (temp[i]=='0') digitalWrite(8, LOW);
9        if (temp[i]=='1') digitalWrite(8, HIGH);
10     }
11   }
12 }
```

Programska koda 4.13: Vzpostavljanje povezave Wi-Fi Direct

```
uint8_t P2PClass::connect() {
  flush();
  Serial.println(F("AT+WRXACTIVE=1")); // izklop varčevanja energije
  if (!checkResponse())
    return 0;

  flush();
  Serial.print(F("AT+NSET="));          // naslovi IP
  Serial.print(server_addr);
  Serial.print(F(",255.255.255.0,"));
  Serial.println(server_addr);
  if (!checkResponse())
    return 0;

  flush();
  Serial.println(F("AT+P2PSETDEV=15,81,1,6,430C,FR"));// število o lastniku je 15
  if (!checkResponse())
    return 0;

  flush();
  Serial.print(F("AT+P2PSETWPS="));
  Serial.print(name);                   // ime naprave vidno pri iskanju
  Serial.println(F(",000a,0005,11223344556677881122334455667733"));
  if (!checkResponse())
    return 0;

  flush();
  Serial.println(F("AT+WM=3")); // način Wi-Fi Direct
  if (!checkResponse())
    return 0;

  flush();
  Serial.println(F("AT+P2PLISTEN=,"));// čakanje na zahtevo za prikaz števila PIN
  if (!checkResponse())
    return 0;
  String buf;
  buf = readline();
  while (!buf.startsWith("p2p-prov-disc-req display-pin"))
    buf = readline();

  flush();
  Serial.print(F("AT+P2PGRPFORM="));    // ustvarjanje skupine
  Serial.print(buf.substring(40, 57));  // naslov MAC naprave ki zahteva povezavo
  Serial.print(F(",,2,"));
  Serial.print(PIN);                    // PIN
  Serial.println(F(",15,1,1"));
  if (!checkResponse())
    return 0;

  flush();
  Serial.print(F("AT+NSTCP="));  // strežnik
  Serial.println(server_port);   // vrata
  if (!checkResponse())
    return 0;
return 1;
}
```

POGLAVJE 5

Wi-Fi Direct aplikacija Android

Sama povezava Wi-Fi Direct med modulom GainSpan in napravo Android nam ne omogoča pošiljanja sporočil TCP/IP, če se na napravi Android ne vzpostavi odjemalec. Sporočila bi lahko pošiljali z že narejeno aplikacijo iz trgovine aplikacij Android, katere namen je vzpostavljanje povezave s strežnikom in izmenjava sporočil. Pri vzpostavljanju povezave bi morali vnesti naslov in vrata strežnika. Operacijski sistem Android v sistemskem grafičnem vmesniku ne podpira branja informacij o povezavi Wi-Fi Direct, oziroma ni mogoče enostavno dobiti informacije o lastniku skupine, oziroma v našem primeru tudi naslova IP strežnika TCP/IP. Pri vzpostavljanju povezave Wi-Fi Direct pa moramo vsakič vpisovati število PIN.

Da bi se izognili ponavljajočemu vpisovanju števila PIN, naslova IP strežnika in vrat ter vzpostavljanja odjemalca TCP/IP, smo omenjene funkcionalnosti implementirali v Wi-Fi Direct aplikacijo Android, ki je del programskega paketa Android SDK.

5.1 Razvojno okolje aplikacij Android

Priporočeno okolje za razvoj aplikacij Android je sestavljeno iz Android SDK [12] in Eclipse IDE [11] z razširitvijo ADT [3].

5.1.1 Android SDK

Android SDK je skupek javanskih knjižnic, vzorčnih aplikacij in orodij, ki omogočajo razvoj, prevajanje, namestitev ter testiranje aplikacij Android. Arhitekturo platforme Android lahko razdelimo na štiri sklope [5]:

– vgrajene aplikacije,

– aplikacijsko ogrodje,
– knjižnice,
– linux jedro.

Zaradi odprtokodnosti platforme imajo razvijalci poln dostop do istih knjižnic, kot že vgrajene osnovne aplikacije vsake naprave Android. Pri samem razvoju je treba biti nekoliko bolj pazljiv na verzije knjižnic, ker te namreč niso podprte na vseh napravah ali pa so se vmesniki med verzijami spremenili. Omenili bi knjižnico `WifiP2pManager`, ki jo dostopna od API verzije 14, oziroma Androida 4.0 in se uporablja pri povezavah Wi-Fi Direct.

5.1.2 Eclipse IDE in razširitev ADT

Eclipse je odprtokodno razvojno okolje z bogatim grafičnim uporabniškim vmesnikom. Čeprav lahko enostavne programe razvijamo z urejevalniki besedil kot je npr. beležnica (angl. notepad) v operacijskem sistemu Windows, je aplikacije z uporabniškim grafičnim vmesnikom bistveno težje ter časovno zamudno razvijati v le-teh. Eclipse nam tako ponuja več prednosti:

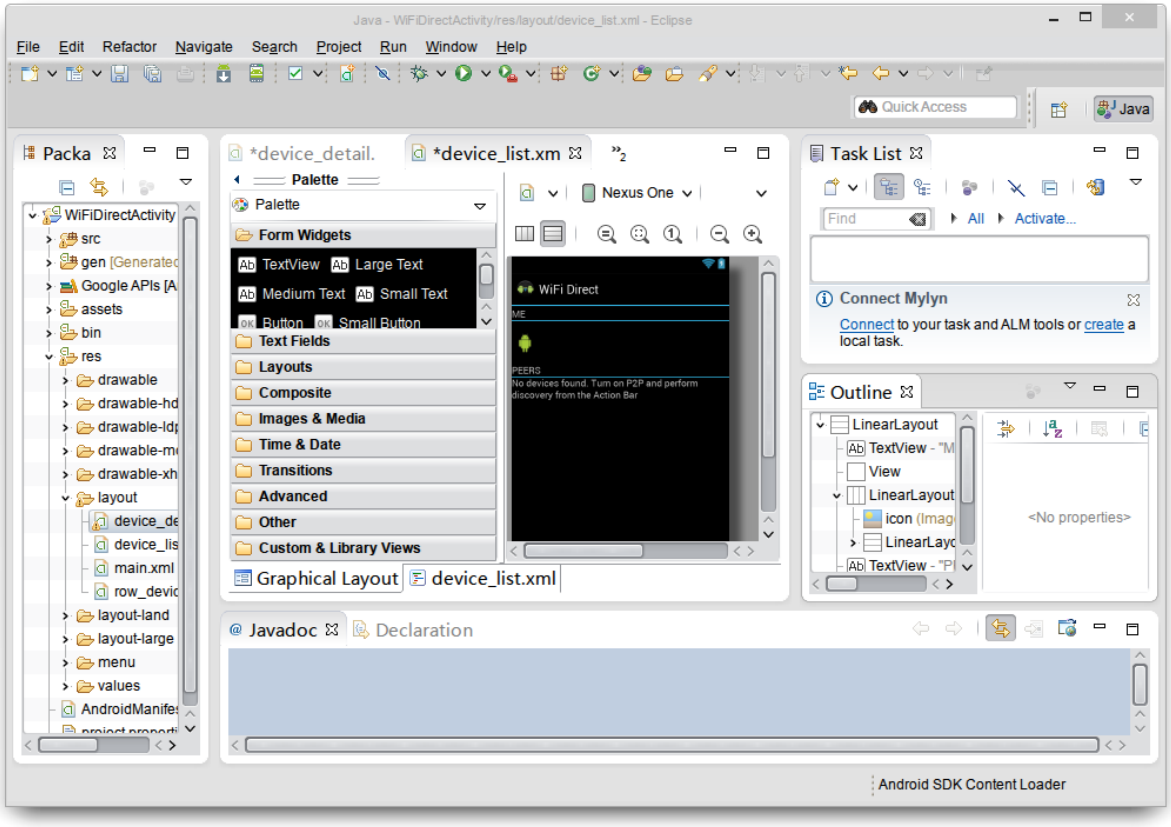
– formatiranje kode,
– sintaktično preverjanje kode,
– razhroščevanje kode,
– avtomatično svetovanje in dokončanje metod razredov,
– prevajanje kode,
– izdelava grafičnega vmesnika aplikacij.

Dodatne funkcionalnosti so implementirane s pomočjo razširitev (angl. plugins). Namestijo se iz samega razvojnega okolja iz množice ponujenih. Izdelavo grafičnega vmesnika, prevajanje in namestitev Android aplikacij podpira ADT razširitev (slika 5.1) ki s pomočjo Android SDK tvori celovito platformo za izdelavo mobilnih aplikacij.

5.2 Razredi Android za vzpostavljanje povezav Wi-Fi Direct

Glavni razred za komunikacijo s sistemsko storitvijo za upravljanje povezav Wi-Fi Direct je razred `WifiP2pManager`. Razred se inicializira z metodo:

■ `(WifiP2pManager)getSystemService(Context.WIFI_P2P_SERVICE);`.



Slika 5.1: Eclipse IDE z razširitvijo ADT

Da bi se aplikacija zavedala, če se izgubi dostop do sistemske storitve, oziroma kanala s pomočjo katerega aplikacija komunicira s sistemsko storitvijo, mora implementirati vmesnik:

- `ChannelListener`,

ter metodo vmesnika:

- `onChannelDisconnected();`,

v kateri se lahko ponovno poskuša povezati s sistemsko storitvijo s ponovno inicializacijo kanala:

- `(Channel)(WifiP2pManager)initialize(this,getMainLooper(),);`.

Metoda za vzpostavljanje povezave Wi-Fi Direct je:

- `(WifiP2pManager)connect((Channel),(WifiP2pConfig),);`,

kjer se z razredom `WifiP2pConfig` definira metoda WPS, naslov MAC naprave s katero se povezuje, število o lastniku skupine in število PIN:

- `WpsInfo.wps.setup` ki sprejema naslednje parametre:
  1. `WpsInfo.KEYPAD`,
  2. `WpsInfo.DISPLAY`,
  3. `WpsInfo.LABEL`,
  4. `WpsInfo.PBC`,
- `deviceAdress`,
- `groupOwnerIntent`,
- `wps.pin`.

Aplikacija za poslušanje dogodka o uspešno vzpostavljeni povezavi mora implementirati vmesnik:

- `ConnectionInfoListener`,

ter metodo:

- `onConnectionInfoAvailable(WifiP2pInfo info)`,

s pomočjo katere v metodi vmesnika z razredom `WifiP2pInfo` dobi informacije o skupini:

- `WifiP2pInfo.groupFormed` – skupina je ustvarjena,
- `WifiP2pInfo.isGroupOwner` – je lastnik skupine,
- `WifiP2pInfo.groupOwnerAddress` – naslov IP lastnika skupine.

Iskanje naprav Wi-Fi Direct je omogočeno z metodo razreda:

- `(WifiP2pManager)discoverPeers((Channel)(WifiP2pManager))`,

aplikacija pa mora implementirati vmesnik:

- `PeerListListener`,

ter metodo:

- `onPeersAvailable(WifiP2pDeviceList)`,

s pomočjo katere pridobi seznam `WifiP2pDeviceList` z informacijami o dostopnih napravah Wi-Fi Direct.

Od aplikacije je odvisno na kakšen način se bo seznam prikazal, informacije o posamezni napravi pa se hranijo v razredu `WiFiP2pDevice` z naslednjimi spremenljivkami:

- `deviceAddress` – naslov MAC ki ga uporabimo pri povezavi,
- `deviceName` – ime naprave,
- `primaryDeviceType`,
- `secondaryDeviceType`,
- `status`.

5.3 Povezovanje z metodo WPS vpiši

Aplikacija Wi-Fi Direct je za povezovanje uporabljala metodo WPS tipka. Ko smo pri metodi povezovanja definirali število PIN, metodo WPS prikaži in število o lastniku skupine:

- `(WifiP2pConfig)wps.pin = ”12345670”;`,
- `(WifiP2pConfig)wps.setup = WpsInfo.DISPLAY;`,
- `(WifiP2pConfig)config.groupOwnerIntent = 0;`,

se je pri povezovanju vseeno prikazalo dinamično generirano število PIN. Če je bila definirana metoda:

- `(WifiP2pConfig)wps.setup = WpsInfo.KEYPAD;`,

je naprava Android brez kakršnegakoli pogovornega okna začela s povezavo, modul GainSpan je normalno prikazal, da se zahteva povezava s prikazom števila PIN. Ko smo podali ukaz za povezovanje s prikazom in enakim številom PIN kot pri napravi Android, se je skupina normalno ustvarila in naprave so se povezale. Druga dobra lastnost povezovanja z aplikacijo je ta, da ko je skupina že ustvarjena, se je naprava Android brez težav pridružila skupini.

5.4 Vzpostavljanje odjemalca TCP/IP

Aplikacija je podpirala prenos datotek med napravami. Ker teh razredov nismo mogli enostavno predelati za naše potrebe, smo ustvarili novega. Nov razred ustvari povezavo s strežnikom po vzpostavitvi skupine in pridobitvi informacij o lastniku skupine, oziroma naslovu strežnika. Operacijski sistem Android ne dopušča zank v glavni niti (angl. thread) aplikacij, zaradi tega smo morali branje sporočil narediti v zasebni niti. Obstaja več Android in javanskih razredov ki omogočajo ustvarjanje novih niti, odločili pa smo se za uporabo razreda

`AsyncTask`. Zanka mora teči v metodi `doInBackground(Void... params)`. V zanki se preverjajo sporočila TCP/IP, ko je sporočilo dostopno za branje se uporabi funkcija `publishProgress(sporočilo);` s pomočjo katere se sporočilo prenese metodi `onProgressUpdate(byte[]...values)`, ta pa lahko, kot v našem primeru v tekstovno polje izpiše novo sporočilo. Ko se zanka konča, oziroma ko se povezava prekine, razred `AsyncTask` izvede metodo `onCancelled()`. V tej metodi se kliče npr. funkcija aplikacije ki odloči kaj se zgodi po prekinitvi povezave.

Android aplikacije se po rotaciji zaslona znova zaženejo in inicializirajo. Povezava Wi-Fi Direct je še vedno bila vzpostavljena, ker se njene informacije pridobijo iz sistemskih storitev, povezava TCP/IP pa se je prekinila in na novo vzpostavila. Modul GainSpan je povezave pred rotacijo še vedno zaznaval kot da so povezane. Ker modul podpira samo 15 povezav TCP/IP, bi le-teh kar hitro zmanjkalo. Rešitev je bila, da razred za branje sporočil razširimo (angl. extends) z razredom `Fragment`. Glavna nit Android aplikacije pri rotaciji preveri če naš razred za branje sporočil obstaja, drugače pa ustvari novega (koda 5.1).

Programska koda 5.1: Klic starega, ali ustvarjanje novega razreda pri rotaciji naprave Android

```
1 FragmentManager fm = getFragmentManager();
2 mTaskFragment = (TaskFragment) fm.findFragmentByTag("task");
3
4 if (mTaskFragment == null) {
5       mTaskFragment = new TaskFragment();
6       fm.beginTransaction().add(mTaskFragment, "task").commit();
7 }
```

5.5 Shranjevanje števila PIN in vrat strežnika

Modul GainSpan ne podpira oglaševanja storitev, zaradi tega je potrebno hraniti številko vrat strežnika, prav tako pa moramo hraniti število PIN. Prikaz pogovornega okna (slika 5.3a) za vnos informacij, ki se prikaže s dotikom na ime naprave (koda 5.2), smo ustvarili z uporabo razreda `AlertDialog.Builder` s tipkami za potrditev in preklic vnosa. Polja za vnos števil so narejena z uporabo razreda `EditText`, v katerem smo dovolili samo vnos številk, združili pa smo jih v linearno postavitev z uporabo razreda `LinearLayout`, ta pa je dodan kot pogled pogovornega okna. Pri potrditvi vnosa se preverja dolžina obeh polj za vnos številk, število PIN mora biti dolžine 8 številk, število vrat strežnika pa mora biti večje od nič. Če so dolžine polj pravilnih dolžin, se s tipko za potrditev shranijo s pomočjo razreda `SharedPreferences.Editor` ter podajo kot parametri povezave Wi-Fi Direct, oziroma povezave TCP/IP odjemalca. Vsako število se hrani s pomočjo ključa, ki je v našem primeru ime naprave Wi-Fi Direct. V primeru ko imamo že shranjene informacije o napravi s katero se

povezujemo, se te preverijo, in če so dolžine pravilne se vzpostavi povezava brez pogovornega okna za vnos številk PIN in vrat strežnika.

Originalna aplikacija je pri dotiku imena naprave prikazala informacije o napravi ter tipko za povezovanje ko ni vzpostavljena povezava, ko pa je bila vzpostavljena, pa je prikazala novo tipko za prekinitev povezave. Ti dve tipki smo združili tako, da se pri dotiku imena vzpostavi povezava če ni vzpostavljena, ter prekine, če je vzpostavljena. S tem smo pridobili prostor za dodatne tipke ali kakšne drugačne grafične gradnike aplikacij.

Programska koda 5.2: Vnos števila PIN in vrat strežnika na napravi Android

```
@Override
public void onListItemClick(ListView l, View v, int position, long id) {

final WifiP2pDevice device = (WifiP2pDevice)getListAdapter().getItem(position);


final SharedPreferences sp = getActivity().getSharedPreferences(
   getResources().getString(R.string.PeerPINS), Activity.MODE_PRIVATE);

final SharedPreferences sp2 = getActivity().getSharedPreferences(
   getResources().getString(R.string.PeerPORTS), Activity.MODE_PRIVATE);

final SharedPreferences.Editor editor = sp.edit();
final SharedPreferences.Editor editor2 = sp2.edit();

final WifiP2pConfig config =  new WifiP2pConfig();
config.deviceAddress = device.deviceAddress;
config.wps.setup = WpsInfo.KEYPAD; //pin se vpiše, imamo shranjeno
//config.wps.setup = WpsInfo.DISPLAY;//pin se prikaže na androidu
//config.wps.setup = WpsInfo.LABEL; //gainspan ne zazna
//config.wps.setup = WpsInfo.PBC;
config.groupOwnerIntent = 0;

if (device.status==WifiP2pDevice.CONNECTED ||
   device.status==WifiP2pDevice.INVITED)
      ((DeviceActionListener) getActivity()).cancelDisconnect();

else if (sp.getString(device.deviceName, null)==null ||
         sp2.getString(device.deviceName, null)==null ) {

   final EditText input = new EditText(getActivity());
   final EditText input2 = new EditText(getActivity());
   input.setHint("8 digit pin");
   input2.setHint("server port");
   input.setInputType(InputType.TYPE_CLASS_NUMBER);
   input2.setInputType(InputType.TYPE_CLASS_NUMBER);

    LinearLayout ll=new LinearLayout(getActivity());
    ll.setOrientation(LinearLayout.VERTICAL);
    ll.addView(input);
    ll.addView(input2);
```

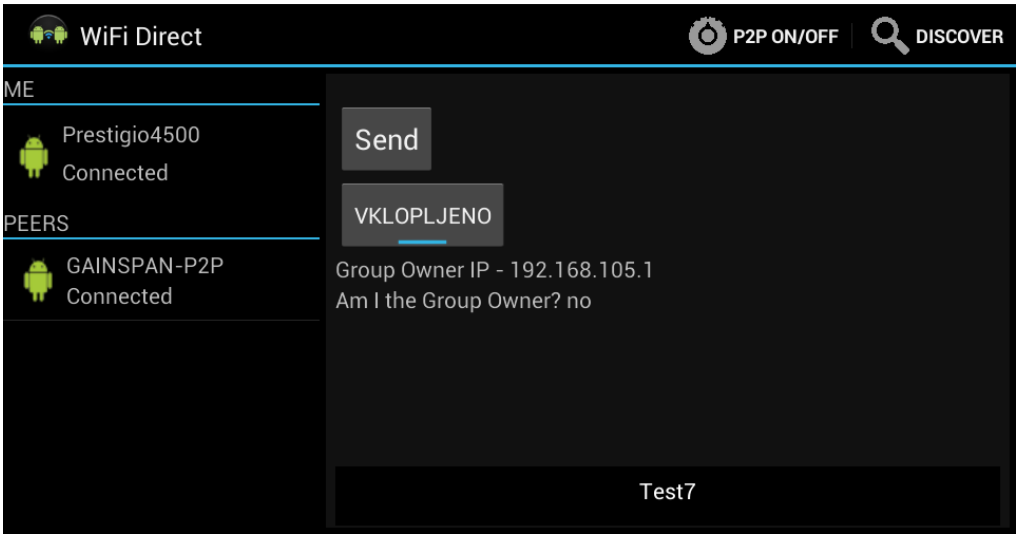
```
42
43      new AlertDialog.Builder(getActivity())
44        .setTitle("Enter " +device.deviceName+ " PIN and server port:")
45        .setView(ll)
46        .setPositiveButton(android.R.string.yes,
47           new DialogInterface.OnClickListener() {
48              public void onClick(DialogInterface dialog, int which) {
49                 if (input.getText().length()==8 &&
50                     input2.getText().length()>0) {
51
52                    config.wps.pin = input.getText().toString();
53                    editor.putString(device.deviceName,
54                        config.wps.pin.toString());
55
56                    editor2.putString(device.deviceName,
57                        input2.getText().toString());
58
59                    editor.commit();
60                    editor2.commit();
61
62                    ((DeviceActionListener)getActivity()).connect(config);
63                    mTaskFragment.port=Integer.parseInt(
64                       sp2.getString(device.deviceName, null));
65                 }
66              }
67           })
68        .setNegativeButton(android.R.string.no,
69           new DialogInterface.OnClickListener() {
70              public void onClick(DialogInterface dialog, int which) {}
71           })
72        .setIcon(android.R.drawable.ic_dialog_info)
73        .show();
74     }
75  else {
76     config.wps.pin=sp.getString(device.deviceName, null);
77     mTaskFragment.port=Integer.parseInt(
78        sp2.getString(device.deviceName, null));
79     ((DeviceActionListener) getActivity()).connect(config);
80     }
81
82  }
```

Brisanje in izpis informacij o napravi (koda 5.3) Wi-Fi Direct smo naredili tako, da se z daljšim dotikom na ime najdene naprave prikaže pogovorno okno z dvemi vrsticami. Z izbiro vrstice `Info` se prikaže pogovorno okno z informacijami o napravi, z izbiro vrstice `Delete PIN and server port` pa se izbrišejo shranjene nastavitve povezave – število PIN in vrata strežnika (slika 5.3b).

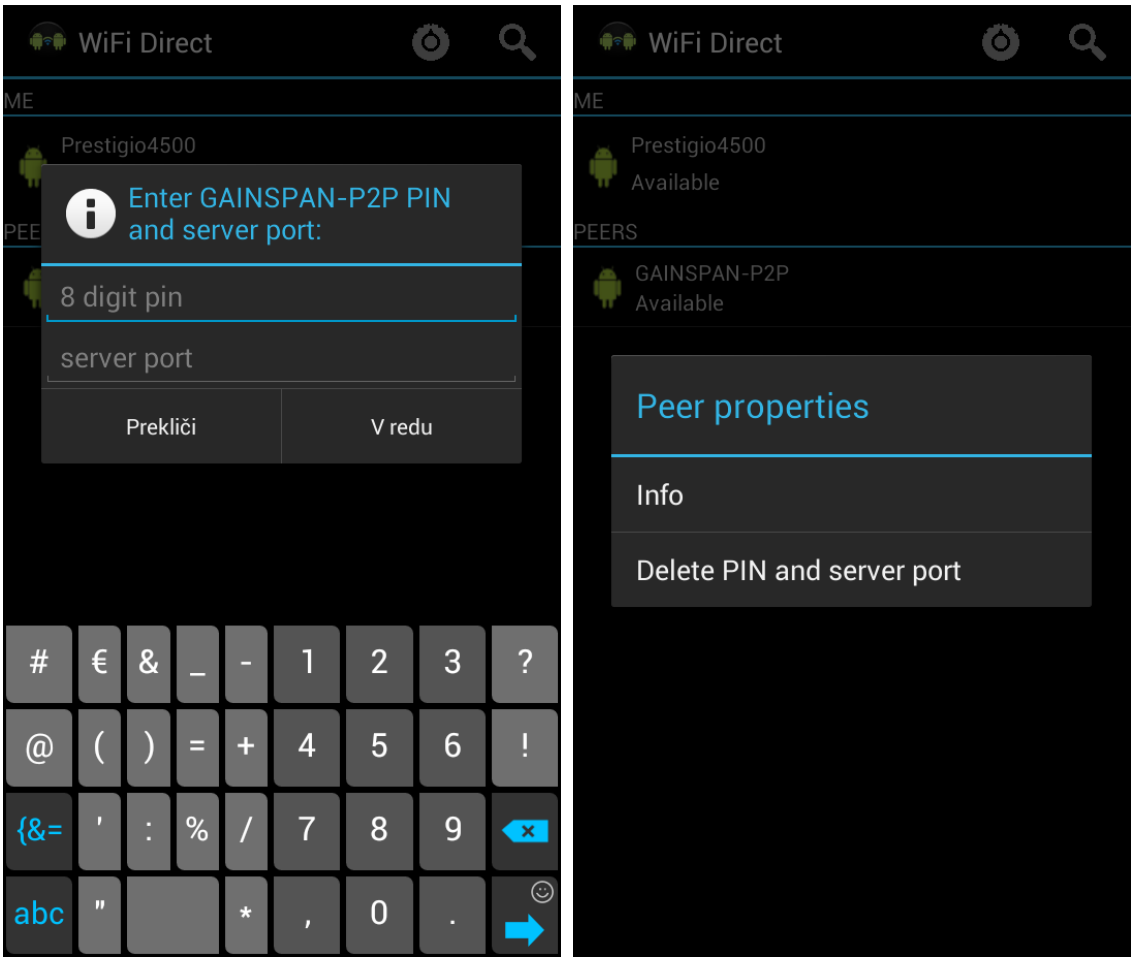
Programska koda 5.3: Brisanje števila PIN in vrat strežnika

```
@Override
public void onCreateContextMenu(ContextMenu menu,View v,
   ContextMenuInfo menuInfo){
      super.onCreateContextMenu(menu, v, menuInfo);
      menu.setHeaderTitle("Peer properties");
      menu.add(Menu.NONE,Menu.NONE,Menu.NONE,"Info");
      menu.add(Menu.NONE,Menu.NONE,Menu.NONE,
         "Delete PIN and server port");
}

@Override
public boolean onContextItemSelected(MenuItem item) {

AdapterContextMenuInfo info=
   (AdapterContextMenuInfo)item.getMenuInfo();

if (item.toString().equals("Info")) {
   AlertDialog.Builder builder =
   new AlertDialog.Builder(getActivity());

   builder.setMessage(getListAdapter().getItem(info.position)
      .toString()).setTitle("Peer info");
   AlertDialog dialog = builder.create();
   dialog.show();
}
else if (item.toString().equals("Delete PIN and server port")) {
   SharedPreferences sp = getActivity().getSharedPreferences(
   getResources().getString(R.string.PeerPINS),Activity.MODE_PRIVATE);

   SharedPreferences.Editor editor = sp.edit();
   editor.remove(((WifiP2pDevice)getListAdapter()
      .getItem(info.position)).deviceName);
   editor.commit();

   SharedPreferences sp2 = getActivity().getSharedPreferences(
   getResources().getString(R.string.PeerPORTS),Activity.MODE_PRIVATE);

   SharedPreferences.Editor editor2 = sp2.edit();
   editor2.remove(((WifiP2pDevice)getListAdapter()
     .getItem(info.position)).deviceName);
   editor2.commit();
   }
   return super.onContextItemSelected(item);
}
```

Dve dodani tipki (slika 5.2) so namenjeni testiranju prenosa sporočil TCP/IP. Tipka `Send` pošlje sporočilo z besedo `Test` kateremu je dodano naključno število od 0 do 9. Tipka za vklop LED diode ima dve stanji, ki se preverjajo, če je vklopljena pošlje črko 0, drugače pa 1. Uporabniški mikrokrmilni program preveri sporočilo in v primeru ničle ugasne LED diodo, v primeru enke pa vklopi. Sporočila se izpisujejo v tekstovno polje pri dnu aplikacije, v našem testnem primeru pa strežnik vrača sprejeta sporočila.



Slika 5.2: Pošiljanje testnih sporočil



(a) Pogovorno okno za vnos številk PIN in vrat strežnika, ki se prikaže v primeru, ko se prvič povezujemo z napravo ali po brisanju

(b) Pogovorno okno za brisanje ter prikaz informacij o napravi, ki se prikaže pri daljšem dotiku imena naprave

Slika 5.3: Dodana pogovorna okna aplikacije Wi-Fi Direct

POGLAVJE 6

Sklepne ugotovitve

V diplomskem delu smo na splošno predstavili brezžična omrežja standarda IEEE 802.11. Bolj podrobno smo opisali standard Wi-Fi Direct ter vse potrebne korake pri vzpostavljanju povezav Wi-Fi Direct med računalnikom, napravo Android in modulom GainSpan 1500M.

Uspešno smo rešili problematiko avtomatičnega vzpostavljanja povezave Wi-Fi Direct z modulom s pomočjo mikrokrmilne knjižnice in mikrokrmilne platforme Arduino UNO. Rešili smo tudi problematiko sprejemanja in pošiljanja sporočil TCP/IP ter tako omogočili mikrokrmilniku, da na podlagi sporočil TCP/IP krmili tudi druge periferne naprave.

Aplikacijo Wi-Fi Direct smo nadgradili na način, da omogoča avtomatično vzpostavljanje povezave Wi-Fi Direct s pomočjo shranjenih informacij o vsaki napravi. Po vzpostavitvi povezave Wi-Fi Direct se avtomatično vzpostavi odjemalec TCP/IP, s pomočjo katerega lahko pošiljamo in sprejemamo sporočila oddaljenega vgrajenega sistema.

Aplikacija Wi-Fi Direct, knjižnica Arduino, Arduino UNO in modul GainSpan 1500M predstavljajo osnovne gradnike za upravljanje in zaznavanje oddaljene vgrajene naprave. Prednost uporabe takšne vgrajene naprave je, da deluje kjerkoli ter ni odvisna od infrastrukturnega načina povezave, oziroma brezžične dostopne točke. Uporabniku, ki je povezan z napravo pa omogoča neovirano uporabo infrastrukturnega načina povezave.

V procesu izvedbe diplomskega dela je bila potrebna precejšnja mera samostojnosti. Potrebno si je bilo vzeti čas in naštudirati teoretični del, ki vključuje branje dokumentacij mnogih uporabljenih standardov, protokolov, načinov uporabe javanskih in Arduino knjižnic, kot tudi spoznati sam proces izdelave aplikacij Android ter življenjski cikel le-teh. Bilo je potrebno obnoviti ter nadgraditi teoretično znanje, ki smo ga pridobili tekom študija.

Pri praktični izvedbi diplomskega dela je bilo potrebno izvajati mnogo testiranja in odkrivanja napak. Presenečeni smo bili nad množico nepričakovanih ovir, ki jih je bilo potrebno

premagati. Menim pa, da je ravno to tisto kar človeka izpopolni in mu pomaga razširiti njegovo praktično inženirsko znanje.

Kazalo slik

Kazalo programske kode

Kazalo tabel

Literatura

[1] Wiring, an open-source programming framework for microcontrollers. Zadnji dostop, dne 03.05.2014, na:
http://wiring.org.co/

[2] Processing Environment (IDE). Zadnji dostop, dne 05.06.2014, na:
http://www.processing.org/

[3] Android Development Tools. Zadnji dostop, dne 01.07.2014, na:
http://developer.android.com/tools/sdk/eclipse-adt.html

[4] WPA Supplicant–Wikipedia. Zadnji dostop, dne 25.05.2014, na:
http://en.wikipedia.org/wiki/Wpa_supplicant

[5] Android–Wikipedia. Zadnji dostop, dne 15.06.2014, na:
http://en.wikipedia.org/wiki/Android_(operating_system)

[6] IEEE 802.11n. Zadnji dostop, dne 20.06.2014, na:
http://www.vocal.com/networking/ieee-802-11n/

[7] Wireless LAN Standards and Topologies. Zadnji dostop, dne 21.06.2014, na:
http://www.mhprofessional.com/downloads/products/0071701524/0071701524_chap02.pdf

[8] Serial to Wi-Fi command reference. Zadnji dostop, dne 20.05.2014, na:
https://s3.amazonaws.com/site_support/uploads/document_upload/Serial_to_WiFi_Command_Reference_5_13.pdf

[9] Wi-Fi Peer-to-Peer (P2P) Technical Specification. Version 1.2

[10] Nmap Security Scanner. Zadnji dostop, dne 15.06.2014, na:
http://nmap.org/

[11] Eclipse. Zadnji dostop, dne 18.06.2014, na:
https://www.eclipse.org/

[12] Android SDK. Zadnji dostop, dne 19.06.2014, na:
http://developer.android.com/sdk/index.html

[13] Enabling comprehensive backward compatibility in the WLAN – without degrading network performance. Zadnji dostop, dne 20.06.2014, na:
http://www.motorolasolutions.com/web/Business/_Documents/Technical_Briefs/_Static_Files/WiFi_Backward_Compatability_In_WLAN.pdf